LAMPIRAN A

FOTO ALAT

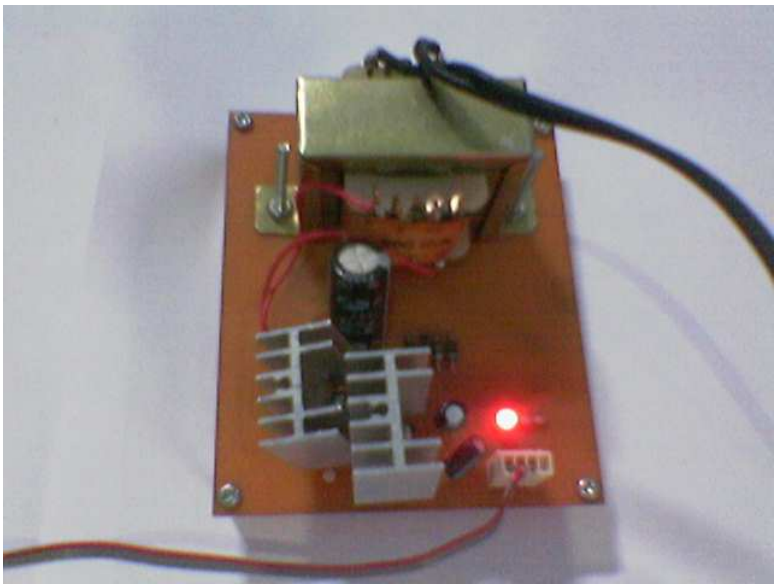


Foto 1 Rangkaian Regulator

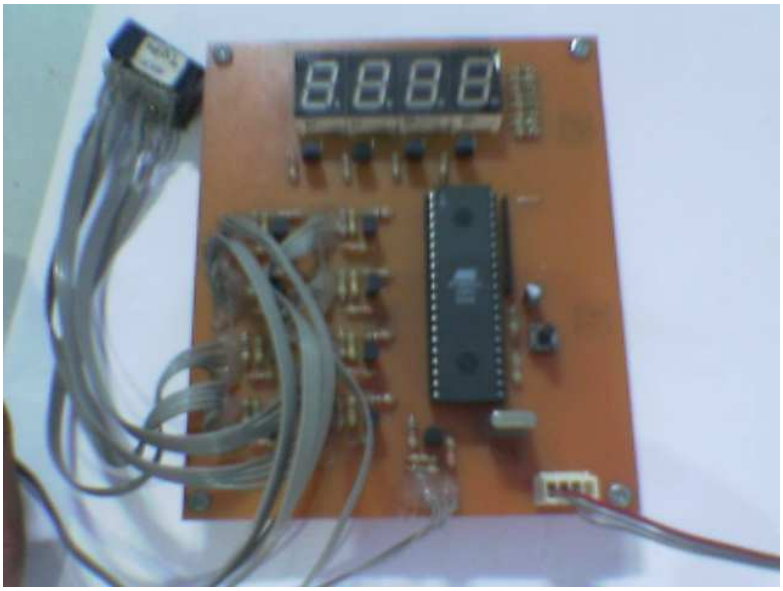


Foto 2 Alat Pencatat Tarif Biaya Angkot

Features

- Compatible with MCS-51™ Products
- 4K Bytes of In-System Reprogrammable Flash Memory
  - Endurance: 1,000 Write/Erase Cycles
- Fully Static Operation: 0 Hz to 24 MHz
- Three-Level Program Memory Lock
- 128 x 8-Bit Internal RAM
- 32 Programmable I/O Lines
- Two 16-Bit Timer/Counters
- Six Interrupt Sources
- Programmable Serial Channel
- Low Power Idle and Power Down Modes

Description

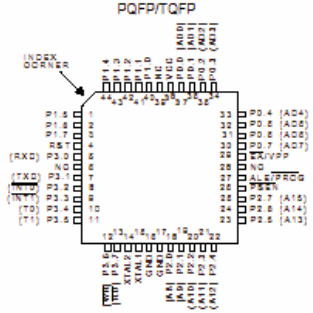
The AT89C51 is a low-power, high-performance CMOS 8-bit microcomputer with 4K bytes of Flash Programmable and Erasable Read Only Memory (PEROM). The device is manufactured using Atmel's high density nonvolatile memory technology and is compatible with the industry standard MCS-51™ instruction set and pinout. The on-chip Flash allows the program memory to be reprogrammed in-system or by a conventional nonvolatile memory programmer. By combining a versatile 8-bit CPU with Flash on a monolithic chip, the Atmel AT89C51 is a powerful microcomputer which provides a highly flexible and cost effective solution to many embedded control applications.

(continued)

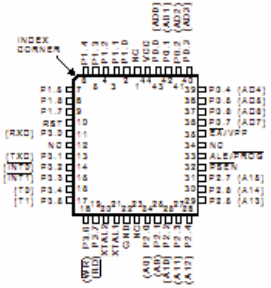
Pin Configurations

PDIP

PQFP/TQFP



PLCC

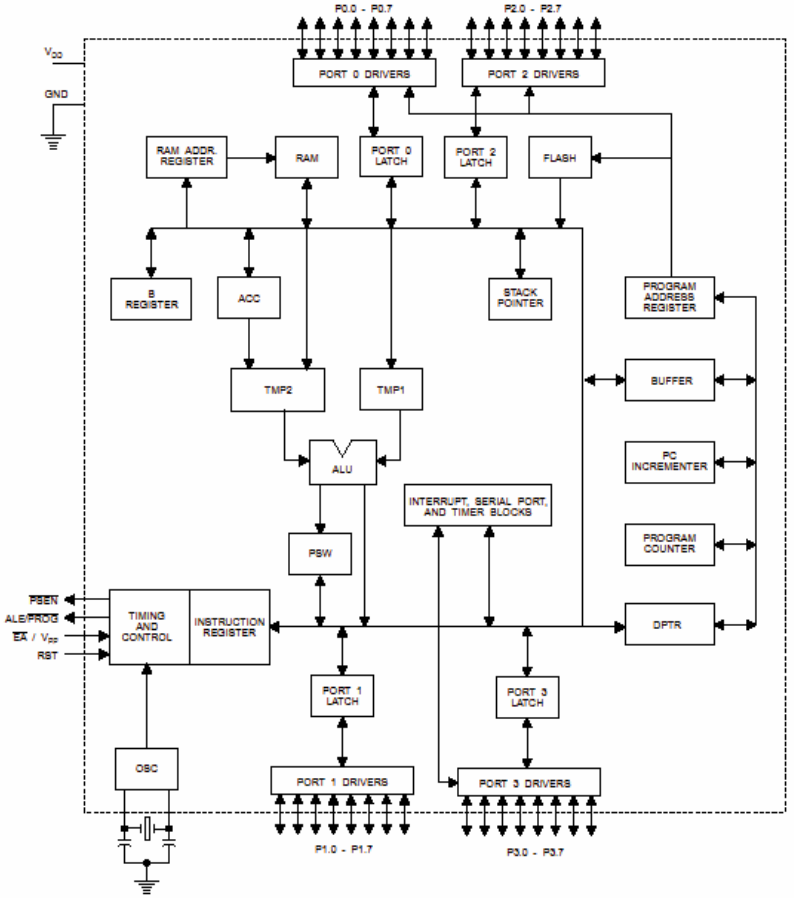


8-Bit Microcontroller with 4K Bytes Flash

AT89C51

0265F-A–12/97

Block Diagram



The AT89C51 provides the following standard features: 4K bytes of Flash, 128 bytes of RAM, 32 I/O lines, two 16-bit timer/counters, a five vector two-level interrupt architecture, a full duplex serial port, on-chip oscillator and clock circuitry. In addition, the AT89C51 is designed with static logic for operation down to zero frequency and supports two software selectable power saving modes. The Idle Mode stops the CPU while allowing the RAM, timer/counters, serial port and interrupt system to continue functioning. The Power Down Mode saves the RAM contents but freezes the oscillator disabling all other chip functions until the next hardware reset.

Pin Description

V_{CC}
Supply voltage.

GND
Ground.

Port 0
Port 0 is an 8-bit open drain bidirectional I/O port. As an output port each pin can sink eight TTL inputs. When 1s are written to port 0 pins, the pins can be used as high-impedance inputs.

Port 0 may also be configured to be the multiplexed low-order address/data bus during accesses to external program and data memory. In this mode P0 has internal pullups.

Port 0 also receives the code bytes during Flash programming, and outputs the code bytes during program verification. External pullups are required during program verification.

Port 1
Port 1 is an 8-bit bidirectional I/O port with internal pullups. The Port 1 output buffers can sink/source four TTL inputs. When 1s are written to Port 1 pins they are pulled high by the internal pullups and can be used as inputs. As inputs, Port 1 pins that are externally being pulled low will source current (I_{IL}) because of the internal pullups.

Port 1 also receives the low-order address bytes during Flash programming and verification.

Port 2
Port 2 is an 8-bit bidirectional I/O port with internal pullups. The Port 2 output buffers can sink/source four TTL inputs. When 1s are written to Port 2 pins they are pulled high by the internal pullups and can be used as inputs. As inputs, Port 2 pins that are externally being pulled low will source current (I_{IL}) because of the internal pullups.

Port 2 emits the high-order address byte during fetches from external program memory and during accesses to external data memory that use 16-bit addresses (MOVX @ DPTR). In this application it uses strong internal pullups when emitting 1s. During accesses to external data memory that use 8-bit addresses (MOVX @ RI), Port 2 emits the contents of the P2 Special Function Register.

Port 2 also receives the high-order address bits and some control signals during Flash programming and verification.

Port 3
Port 3 is an 8-bit bidirectional I/O port with internal pullups. The Port 3 output buffers can sink/source four TTL inputs. When 1s are written to Port 3 pins they are pulled high by the internal pullups and can be used as inputs. As inputs, Port 3 pins that are externally being pulled low will source current (I_{IL}) because of the pullups.

Port 3 also serves the functions of various special features of the AT89C51 as listed below:

Port Pin	Alternate Functions
P3.0	RXD (serial input port)
P3.1	TXD (serial output port)
P3.2	$\overline{INT0}$ (external interrupt 0)
P3.3	$\overline{INT1}$ (external interrupt 1)
P3.4	T0 (timer 0 external input)
P3.5	T1 (timer 1 external input)
P3.6	$\overline{WR}$ (external data memory write strobe)
P3.7	$\overline{RD}$ (external data memory read strobe)

Port 3 also receives some control signals for Flash programming and verification.

RST
Reset input. A high on this pin for two machine cycles while the oscillator is running resets the device.

ALE/$\overline{PROG}$
Address Latch Enable output pulse for latching the low byte of the address during accesses to external memory. This pin is also the program pulse input ($\overline{PROG}$) during Flash programming.

In normal operation ALE is emitted at a constant rate of 1/6 the oscillator frequency, and may be used for external timing or clocking purposes. Note, however, that one ALE pulse is skipped during each access to external Data Memory.

If desired, ALE operation can be disabled by setting bit 0 of SFR location 8EH. With the bit set, ALE is active only during a MOVX or MOVC instruction. Otherwise, the pin is weakly pulled high. Setting the ALE-disable bit has no effect if the microcontroller is in external execution mode.

$\overline{PSEN}$
Program Store Enable is the read strobe to external program memory.

When the AT89C51 is executing code from external program memory, $\overline{PSEN}$ is activated twice each machine cycle, except that two $\overline{PSEN}$ activations are skipped during each access to external data memory.

$\overline{EA}$/V_{PP}
External Access Enable. $\overline{EA}$ must be strapped to GND in order to enable the device to fetch code from external program memory locations starting at 0000H up to FFFFH. Note, however, that if lock bit 1 is programmed, $\overline{EA}$ will be internally latched on reset.

$\overline{EA}$ should be strapped to V_{CC} for internal program executions.

This pin also receives the 12-volt programming enable voltage (V_{PP}) during Flash programming, for parts that require 12-volt V_{PP}.

XTAL1
Input to the inverting oscillator amplifier and input to the internal clock operating circuit.

XTAL2
Output from the inverting oscillator amplifier.

Oscillator Characteristics

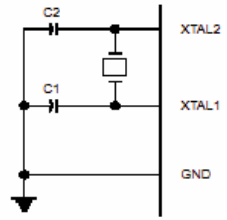
XTAL1 and XTAL2 are the input and output, respectively, of an inverting amplifier which can be configured for use as an on-chip oscillator, as shown in Figure 1. Either a quartz crystal or ceramic resonator may be used. To drive the device from an external clock source, XTAL2 should be left unconnected while XTAL1 is driven as shown in Figure 2. There are no requirements on the duty cycle of the external clock signal, since the input to the internal clocking circuitry is through a divide-by-two flip-flop, but minimum and maximum voltage high and low time specifications must be observed.

Idle Mode

In idle mode, the CPU puts itself to sleep while all the on-chip peripherals remain active. The mode is invoked by software. The content of the on-chip RAM and all the special functions registers remain unchanged during this mode. The idle mode can be terminated by any enabled interrupt or by a hardware reset.

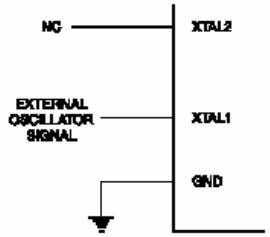
It should be noted that when idle is terminated by a hard ware reset, the device normally resumes program execution, from where it left off, up to two machine cycles before the internal reset algorithm takes control. On-chip hardware inhibits access to internal RAM in this event, but access to the port pins is not inhibited. To eliminate the possibility of an unexpected write to a port pin when Idle is terminated by reset, the instruction following the one that invokes Idle should not be one that writes to a port pin or to external memory.

Figure 1. Oscillator Connections



Note: C1, C2 = 30 pF ± 10 pF for Crystals
= 40 pF ± 10 pF for Ceramic Resonators

Figure 2. External Clock Drive Configuration



Status of External Pins During Idle and Power Down Modes

Mode	Program Memory	ALE	$\overline{PSEN}$	PORT0	PORT1	PORT2	PORT3
Idle	Internal	1	1	Data	Data	Data	Data
Idle	External	1	1	Float	Data	Address	Data
Power Down	Internal	0	0	Data	Data	Data	Data
Power Down	External	0	0	Float	Data	Data	Data

Power Down Mode

In the power down mode the oscillator is stopped, and the instruction that invokes power down is the last instruction executed. The on-chip RAM and Special Function Registers retain their values until the power down mode is terminated. The only exit from power down is a hardware reset. Reset redefines the SFRs but does not change the on-chip RAM. The reset should not be activated before V_{CC} is restored to its normal operating level and must be held active long enough to allow the oscillator to restart and stabilize.

Program Memory Lock Bits

On the chip are three lock bits which can be left unprogrammed (U) or can be programmed (P) to obtain the additional features listed in the table below:

When lock bit 1 is programmed, the logic level at the $\overline{EA}$ pin is sampled and latched during reset. If the device is powered up without a reset, the latch initializes to a random value, and holds that value until reset is activated. It is necessary that the latched value of $\overline{EA}$ be in agreement with the current logic level at that pin in order for the device to function properly.

Lock Bit Protection Modes

Program Lock Bits				Protection Type
	LB1	LB2	LB3	
1	U	U	U	No program lock features.
2	P	U	U	MOVC instructions executed from external program memory are disabled from fetching code bytes from internal memory, $\overline{EA}$ is sampled and latched on reset, and further programming of the Flash is disabled.
3	P	P	U	Same as mode 2, also verify is disabled.
4	P	P	P	Same as mode 3, also external execution is disabled.

Programming the Flash

The AT89C51 is normally shipped with the on-chip Flash memory array in the erased state (that is, contents = FFH) and ready to be programmed. The programming interface accepts either a high-voltage (12-volt) or a low-voltage (V_{CC}) program enable signal. The low voltage programming mode provides a convenient way to program the AT89C51 inside the user's system, while the high-voltage programming mode is compatible with conventional third party Flash or EPROM programmers.

The AT89C51 is shipped with either the high-voltage or low-voltage programming mode enabled. The respective top-side marking and device signature codes are listed in the following table.

	V_{PP} = 12V	V_{PP} = 5V
Top-Side Mark	AT89C51 xxxx yyww	AT89C51 xxxx-5 yyww
Signature	(030H)=1EH (031H)=51H (032H)=FFH	(030H)=1EH (031H)=51H (032H)=05H

The AT89C51 code memory array is programmed byte-by-byte in either programming mode. *To program any non-blank byte in the on-chip Flash Memory, the entire memory must be erased using the Chip Erase Mode.*

Programming Algorithm: Before programming the AT89C51, the address, data and control signals should be set up according to the Flash programming mode table and Figures 3 and 4. To program the AT89C51, take the following steps.

1. Input the desired memory location on the address lines.
2. Input the appropriate data byte on the data lines.
3. Activate the correct combination of control signals.
4. Raise $\overline{EA}/V_{PP}$ to 12V for the high-voltage programming mode.
5. Pulse ALE/$\overline{PROG}$ once to program a byte in the Flash array or the lock bits. The byte-write cycle is self-timed and typically takes no more than 1.5 ms. Repeat steps 1 through 5, changing the address and data for the entire array or until the end of the object file is reached.

Data Polling: The AT89C51 features Data Polling to indicate the end of a write cycle. During a write cycle, an attempted read of the last byte written will result in the complement of the written datum on PO.7. Once the write cycle has been completed, true data are valid on all outputs, and the next cycle may begin. Data Polling may begin any time after a write cycle has been initiated.

Ready/Busy: The progress of byte programming can also be monitored by the RDY/$\overline{BSY}$ output signal. P3.4 is pulled low after ALE goes high during programming to indicate BUSY. P3.4 is pulled high again when programming is done to indicate READY.

Program Verify: If lock bits LB1 and LB2 have not been programmed, the programmed code data can be read back via the address and data lines for verification. The lock bits cannot be verified directly. Verification of the lock bits is achieved by observing that their features are enabled.

Chip Erase: The entire Flash array is erased electrically by using the proper combination of control signals and by holding ALE/$\overline{PROG}$ low for 10 ms. The code array is written with all "1"s. The chip erase operation must be executed before the code memory can be re-programmed.

Reading the Signature Bytes: The signature bytes are read by the same procedure as a normal verification of locations 030H,

031H, and 032H, except that P3.6 and P3.7 must be pulled to a logic low. The values returned are as follows.

(030H) = 1EH indicates manufactured by Atmel
(031H) = 51H indicates 89C51
(032H) = FFH indicates 12V programming
(032H) = 05H indicates 5V programming

Programming Interface

Every code byte in the Flash array can be written and the entire array can be erased by using the appropriate combination of control signals. The write operation cycle is self-timed and once initiated, will automatically time itself to completion.

All major programming vendors offer worldwide support for the Atmel microcontroller series. Please contact your local programming vendor for the appropriate software revision.

Flash Programming Modes

Mode		RST	$\overline{PSEN}$	ALE/$\overline{PROG}$	$\overline{EA}/V_{PP}$	P2.6	P2.7	P3.6	P3.7
Write Code Data		H	L		H/12V	L	H	H	H
Read Code Data		H	L	H	H	L	L	H	H
Write Lock	Bit - 1	H	L		H/12V	H	H	H	H
	Bit - 2	H	L		H/12V	H	H	L	L
	Bit - 3	H	L		H/12V	H	L	H	L
Chip Erase		H	L	 (1)	H/12V	H	L	L	L
Read Signature Byte		H	L	H	H	L	L	L	L

Note: 1. Chip Erase requires a 10-ms PROG pulse.

Figure 3. Programming the Flash

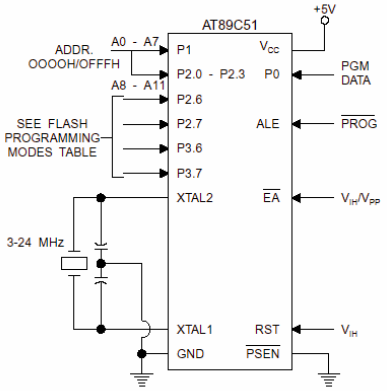
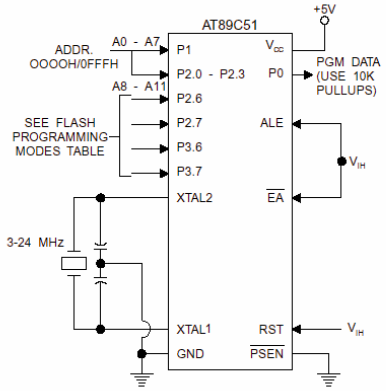


Figure 4. Verifying the Flash



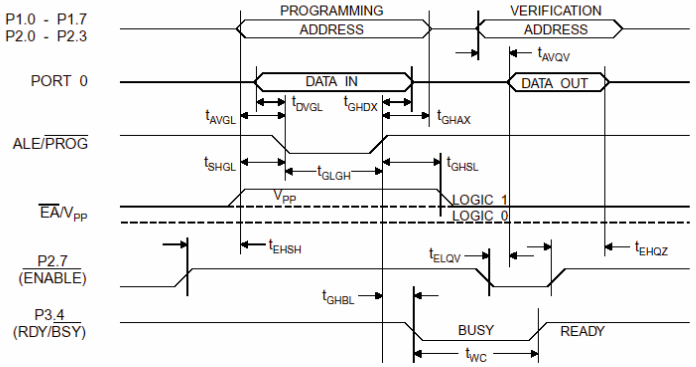
Flash Programming and Verification Characteristics

T_A = 0°C to 70°C, V_{CC} = 5.0 ± 10%

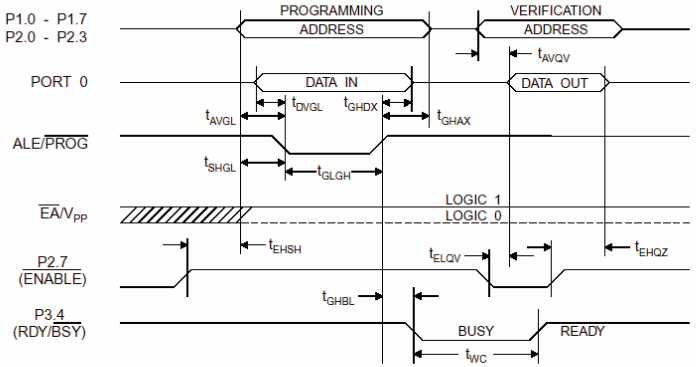
Symbol	Parameter	Min	Max	Units
V_{PP}[1]	Programming Enable Voltage	11.5	12.5	V
I_{PP}[1]	Programming Enable Current		1.0	mA
$1/t_{CLCL}$	Oscillator Frequency	3	24	MHz
t_{AVGL}	Address Setup to $\overline{PROG}$ Low	$48t_{CLCL}$		
t_{GHAX}	Address Hold After $\overline{PROG}$	$48t_{CLCL}$		
t_{DVGL}	Data Setup to $\overline{PROG}$ Low	$48t_{CLCL}$		
t_{GHDX}	Data Hold After $\overline{PROG}$	$48t_{CLCL}$		
t_{EHSH}	P2.7 ($\overline{ENABLE}$) High to V_{PP}	$48t_{CLCL}$		
t_{SHGL}	V_{PP} Setup to $\overline{PROG}$ Low	10		µs
t_{GHSL}[1]	V_{PP} Hold After $\overline{PROG}$	10		µs
t_{GLGH}	$\overline{PROG}$ Width	1	110	µs
t_{AVQV}	Address to Data Valid		$48t_{CLCL}$	
t_{ELQV}	$\overline{ENABLE}$ Low to Data Valid		$48t_{CLCL}$	
t_{EHQZ}	Data Float After $\overline{ENABLE}$	0	$48t_{CLCL}$	
t_{GHBL}	$\overline{PROG}$ High to $\overline{BUSY}$ Low		1.0	µs
t_{WC}	Byte Write Cycle Time		2.0	ms

Note: 1. Only used in 12-volt programming mode.

Flash Programming and Verification Waveforms - High Voltage Mode (V_{PP} = 12V)



Flash Programming and Verification Waveforms - Low Voltage Mode (V_{PP} = 5V)



Absolute Maximum Ratings*

Operating Temperature	-55°C to +125°C
Storage Temperature	-65°C to +150°C
Voltage on Any Pin with Respect to Ground	-1.0V to +7.0V
Maximum Operating Voltage	6.6V
DC Output Current	15.0 mA

*NOTICE: Stresses beyond those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions beyond those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

DC Characteristics

T_A = -40°C to 85°C, V_{CC} = 5.0V ± 20% (unless otherwise noted)

Symbol	Parameter	Condition	Min	Max	Units
V_{IL}	Input Low Voltage	(Except $\overline{EA}$)	-0.5	0.2 V_{CC} - 0.1	V
V_{IL1}	Input Low Voltage ($\overline{EA}$)		-0.5	0.2 V_{CC} - 0.3	V
V_{IH}	Input High Voltage	(Except XTAL1, RST)	0.2 V_{CC} + 0.9	V_{CC} + 0.5	V
V_{IH1}	Input High Voltage	(XTAL1, RST)	0.7 V_{CC}	V_{CC} + 0.5	V
V_{OL}	Output Low Voltage[1] (Ports 1,2,3)	I_{OL} = 1.6 mA		0.45	V
V_{OL1}	Output Low Voltage[1] (Port 0, ALE, $\overline{PSEN}$)	I_{OL} = 3.2 mA		0.45	V
V_{OH}	Output High Voltage (Ports 1,2,3, ALE, $\overline{PSEN}$)	I_{OH} = -60 µA, V_{CC} = 5V ± 10%	2.4		V
		I_{OH} = -25 µA	0.75 V_{CC}		V
		I_{OH} = -10 µA	0.9 V_{CC}		V
V_{OH1}	Output High Voltage (Port 0 in External Bus Mode)	I_{OH} = -800 µA, V_{CC} = 5V ± 10%	2.4		V
		I_{OH} = -300 µA	0.75 V_{CC}		V
		I_{OH} = -80 µA	0.9 V_{CC}		V
I_{IL}	Logical 0 Input Current (Ports 1,2,3)	V_{IN} = 0.45V		-50	µA
I_{TL}	Logical 1 to 0 Transition Current (Ports 1,2,3)	V_{IN} = 2V, VCC = 5V ± 10%		-650	µA
I_{LI}	Input Leakage Current (Port 0, $\overline{EA}$)	0.45 < V_{IN} < V_{CC}		±10	µA
RRST	Reset Pulldown Resistor		50	300	KΩ
C_{IO}	Pin Capacitance	Test Freq. = 1 MHz, T_A = 25°C		10	pF
I_{CC}	Power Supply Current	Active Mode, 12 MHz		20	mA
		Idle Mode, 12 MHz		5	mA
	Power Down Mode[2]	V_{CC} = 6V		100	µA
		V_{CC} = 3V		40	µA

Notes: 1. Under steady state (non-transient) conditions, I_{OL} must be externally limited as follows:
Maximum I_{OL} per port pin: 10 mA
Maximum I_{OL} per 8-bit port: Port 0: 26 mA
Ports 1, 2, 3: 15 mA
Maximum total I_{OL} for all output pins: 71 mA
If I_{OL} exceeds the test condition, V_{OL} may exceed the related specification. Pins are not guaranteed to sink current greater than the listed test conditions.

2. Minimum V_{CC} for Power Down is 2V.

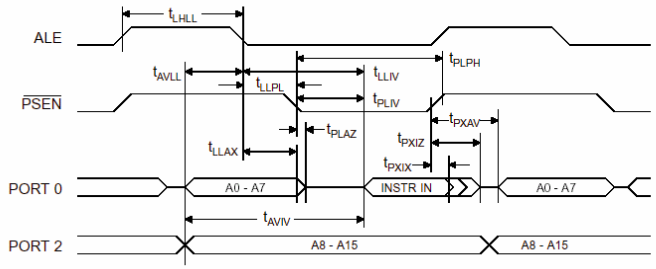
AC Characteristics

(Under Operating Conditions; Load Capacitance for Port 0, ALE/$\overline{PROG}$, and $\overline{PSEN}$ = 100 pF; Load Capacitance for all other outputs = 80 pF)

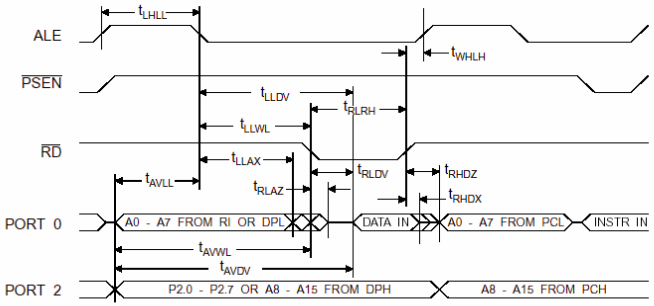
External Program and Data Memory Characteristics

Symbol	Parameter	12 MHz Oscillator		16 to 24 MHz Oscillator		Units
		Min	Max	Min	Max	
$1/t_{CLCL}$	Oscillator Frequency			0	24	MHz
t_{LHLL}	ALE Pulse Width	127		$2t_{CLCL}$-40		ns
t_{AVLL}	Address Valid to ALE Low	43		t_{CLCL}-13		ns
t_{LLAX}	Address Hold After ALE Low	48		t_{CLCL}-20		ns
t_{LLIV}	ALE Low to Valid Instruction In		233		$4t_{CLCL}$-65	ns
t_{LLPL}	ALE Low to $\overline{PSEN}$ Low	43		t_{CLCL}-13		ns
t_{PLPH}	$\overline{PSEN}$ Pulse Width	205		$3t_{CLCL}$-20		ns
t_{PLIV}	$\overline{PSEN}$ Low to Valid Instruction In		145		$3t_{CLCL}$-45	ns
t_{PXIX}	Input Instruction Hold After $\overline{PSEN}$	0		0		ns
t_{PXIZ}	Input Instruction Float After $\overline{PSEN}$		59		t_{CLCL}-10	ns
t_{PXAV}	$\overline{PSEN}$ to Address Valid	75		t_{CLCL}-8		ns
t_{AVIV}	Address to Valid Instruction In		312		$5t_{CLCL}$-55	ns
t_{PLAZ}	$\overline{PSEN}$ Low to Address Float		10		10	ns
t_{RLRH}	$\overline{RD}$ Pulse Width	400		$6t_{CLCL}$-100		ns
t_{WLWH}	$\overline{WR}$ Pulse Width	400		$6t_{CLCL}$-100		ns
t_{RLDV}	$\overline{RD}$ Low to Valid Data In		252		$5t_{CLCL}$-90	ns
t_{RHDX}	Data Hold After $\overline{RD}$	0		0		ns
t_{RHDZ}	Data Float After $\overline{RD}$		97		$2t_{CLCL}$-28	ns
t_{LLDV}	ALE Low to Valid Data In		517		$8t_{CLCL}$-150	ns
t_{AVDV}	Address to Valid Data In		585		$9t_{CLCL}$-165	ns
t_{LLWL}	ALE Low to $\overline{RD}$ or $\overline{WR}$ Low	200	300	$3t_{CLCL}$-50	$3t_{CLCL}$+50	ns
t_{AVWL}	Address to $\overline{RD}$ or $\overline{WR}$ Low	203		$4t_{CLCL}$-75		ns
t_{QVWX}	Data Valid to $\overline{WR}$ Transition	23		t_{CLCL}-20		ns
t_{QVWH}	Data Valid to $\overline{WR}$ High	433		$7t_{CLCL}$-120		ns
t_{WHQX}	Data Hold After $\overline{WR}$	33		t_{CLCL}-20		ns
t_{RLAZ}	$\overline{RD}$ Low to Address Float		0		0	ns
t_{WHLH}	$\overline{RD}$ or $\overline{WR}$ High to ALE High	43	123	t_{CLCL}-20	t_{CLCL}+25	ns

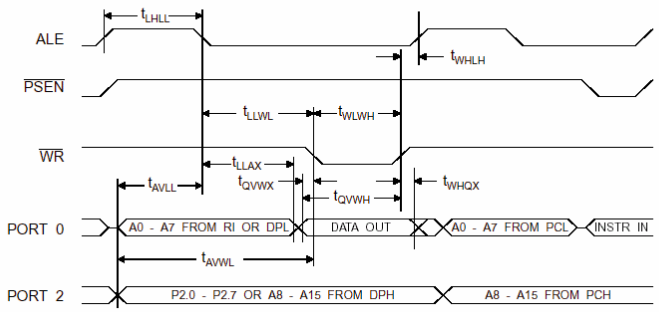
External Program Memory Read Cycle



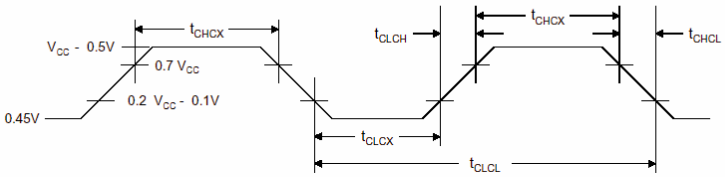
External Data Memory Read Cycle



External Data Memory Write Cycle



External Clock Drive Waveforms



External Clock Drive

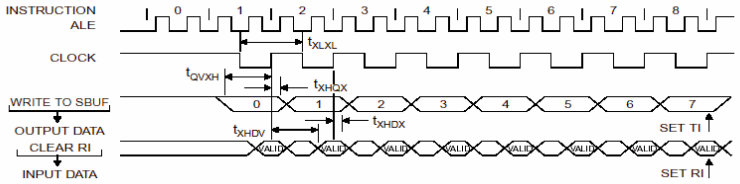
Symbol	Parameter	Min	Max	Units
$1/t_{CLCL}$	Oscillator Frequency	0	24	MHz
t_{CLCL}	Clock Period	41.6		ns
t_{CHCX}	High Time	15		ns
t_{CLCX}	Low Time	15		ns
t_{CLCH}	Rise Time		20	ns
t_{CHCL}	Fall Time		20	ns

Serial Port Timing: Shift Register Mode Test Conditions

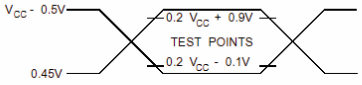
(V_{CC} = 5.0 V ± 20%; Load Capacitance = 80 pF)

Symbol	Parameter	12 MHz Osc		Variable Oscillator		Units
		Min	Max	Min	Max	
t_{XLXL}	Serial Port Clock Cycle Time	1.0		$12t_{CLCL}$		µs
t_{QVXH}	Output Data Setup to Clock Rising Edge	700		$10t_{CLCL}$-133		ns
t_{XHQX}	Output Data Hold After Clock Rising Edge	50		$2t_{CLCL}$-117		ns
t_{XHDX}	Input Data Hold After Clock Rising Edge	0		0		ns
t_{XHDV}	Clock Rising Edge to Input Data Valid		700		$10t_{CLCL}$-133	ns

Shift Register Mode Timing Waveforms

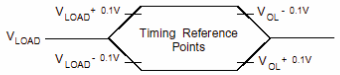


AC Testing Input/Output Waveforms[1]



Note: 1. AC Inputs during testing are driven at V_{CC} - 0.5V for a logic 1 and 0.45V for a logic 0. Timing measurements are made at V_{IH} min. for a logic 1 and V_{IL} max. for a logic 0.

Float Waveforms[1]



Note: 1. For timing purposes, a port pin is no longer floating when a 100 mV change from load voltage occurs. A port pin begins to float when 100 mV change from the loaded V_{OH}/V_{OL} level occurs.

Ordering Information

Speed (MHz)	Power Supply	Ordering Code	Package	Operation Range
12	5V ± 20%	AT89C51-12AC	44A	Commercial
		AT89C51-12JC	44J	(0°C to 70°C)
		AT89C51-12PC	40P6	
		AT89C51-12QC	44Q	
		AT89C51-12AI	44A	Industrial
		AT89C51-12JI	44J	(-40°C to 85°C)
		AT89C51-12PI	40P6	
		AT89C51-12QI	44Q	
		AT89C51-12AA	44A	Automotive
		AT89C51-12JA	44J	(-40°C to 105°C)
		AT89C51-12PA	40P6	
		AT89C51-12QA	44Q	
16	5V ± 20%	AT89C51-16AC	44A	Commercial
		AT89C51-16JC	44J	(0°C to 70°C)
		AT89C51-16PC	40P6	
		AT89C51-16QC	44Q	
		AT89C51-16AI	44A	Industrial
		AT89C51-16JI	44J	(-40°C to 85°C)
		AT89C51-16PI	40P6	
		AT89C51-16QI	44Q	
		AT89C51-16AA	44A	Automotive
		AT89C51-16JA	44J	(-40°C to 105°C)
		AT89C51-16PA	40P6	
		AT89C51-16QA	44Q	
20	5V ± 20%	AT89C51-20AC	44A	Commercial
		AT89C51-20JC	44J	(0°C to 70°C)
		AT89C51-20PC	40P6	
		AT89C51-20QC	44Q	
		AT89C51-20AI	44A	Industrial
		AT89C51-20JI	44J	(-40°C to 85°C)
		AT89C51-20PI	40P6	
		AT89C51-20QI	44Q	

Ordering Information

Speed (MHz)	Power Supply	Ordering Code	Package	Operation Range
24	5V ± 20%	AT89C51-24AC	44A	Commercial
		AT89C51-24JC	44J	(0°C to 70°C)
		AT89C51-24PC	44P6	
		AT89C51-24QC	44Q	
		AT89C51-24AI	44A	Industrial
		AT89C51-24JI	44J	(-40°C to 85°C)
		AT89C51-24PI	44P6	
		AT89C51-24QI	44Q	

Package Type	
44A	44 Lead, Thin Plastic Gull Wing Quad Flatpack (TQFP)
44J	44 Lead, Plastic J-Leaded Chip Carrier (PLCC)
40P6	40 Lead, 0.600" Wide, Plastic Dual Inline Package (PDIP)
44Q	44 Lead, Plastic Gull Wing Quad Flatpack (PQFP)

Microcontroller Instruction Set

For interrupt response time information, refer to the hardware description chapter.

Instructions that Affect Flag Settings(1)

Instruction	Flag			Instruction	Flag		
	C	OV	AC		C	OV	AC
ADD	X	X	X	CLR C	O		
ADDC	X	X	X	CPL C	X		
SUBB	X	X	X	ANL C,bit	X		
MUL	O	X		ANL C,/bit	X		
DIV	O	X		ORL C,bit	X		
DA	X			ORL C,/bit	X		
RRC	X			MOV C,bit	X		
RLC	X			CJNE	X		
SETB C	1						

Note: 1. Operations on SFR byte address 208 or bit addresses 209-215 (that is, the PSW or bits in the PSW) also affect flag settings.

The Instruction Set and Addressing Modes

R_n	Register R7-R0 of the currently selected Register Bank.
direct	8-bit internal data location's address. This could be an Internal Data RAM location (0-127) or a SFR [i.e., I/O port, control register, status register, etc. (128-255)].
@R_i	8-bit internal data RAM location (0-255) addressed indirectly through register R1or R0.
#data	8-bit constant included in instruction.
#data 16	16-bit constant included in instruction.
addr 16	16-bit destination address. Used by LCALL and LJMP. A branch can be anywhere within the 64K byte Program Memory address space.
addr 11	11-bit destination address. Used by ACALL and AJMP. The branch will be within the same 2K byte page of program memory as the first byte of the following instruction.
rel	Signed (two's complement) 8-bit offset byte. Used by SJMP and all conditional jumps. Range is -128 to +127 bytes relative to first byte of the following instruction.
bit	Direct Addressed bit in Internal Data RAM or Special Function Register.

0509B-B–12/97

Instruction Set Summary

	0	1	2	3	4	5	6	7
0	NOP	JBC bit,rel [3B, 2C]	JB bit, rel [3B, 2C]	JNB bit, rel [3B, 2C]	JC rel [2B, 2C]	JNC rel [2B, 2C]	JZ rel [2B, 2C]	JNZ rel [2B, 2C]
1	AJMP (P0) [2B, 2C]	ACALL (P0) [2B, 2C]	AJMP (P1) [2B, 2C]	ACALL (P1) [2B, 2C]	AJMP (P2) [2B, 2C]	ACALL (P2) [2B, 2C]	AJMP (P3) [2B, 2C]	ACALL (P3) [2B, 2C]
2	LJMP addr16 [3B, 2C]	LCALL addr16 [3B, 2C]	RET [2C]	RETI [2C]	ORL dir, A [2B]	ANL dir, A [2B]	XRL dir, a [2B]	ORL C, bit [2B, 2C]
3	RR A	RRC A	RL A	RLC A	ORL dir, #data [3B, 2C]	ANL dir, #data [3B, 2C]	XRL dir, #data [3B, 2C]	JMP @A + DPTR [2C]
4	INC A	DEC A	ADD A, #data [2B]	ADDC A, #data [2B]	ORL A, #data [2B]	ANL A, #data [2B]	XRL A, #data [2B]	MOV A, #data [2B]
5	INC dir [2B]	DEC dir [2B]	ADD A, dir [2B]	ADDC A, dir [2B]	ORL A, dir [2B]	ANL A, dir [2B]	XRL A, dir [2B]	MOV dir, #data [3B, 2C]
6	INC @R0	DEC @R0	ADD A, @R0	ADDC A, @R0	ORL A, @R0	ANL A, @R0	XRL A, @R0	MOV @R0, @data [2B]
7	INC @R1	DEC @R1	ADD A, @R1	ADDC A, @R1	ORL A, @R1	ANL A, @R1	XRL A, @R1	MOV @R1, #data [2B]
8	INC R0	DEC R0	ADD A, R0	ADDC A, R0	ORL A, R0	ANL A, R0	XRL A, R0	MOV R0, #data [2B]
9	INC R1	DEC R1	ADD A, R1	ADDC A, R1	ORL A, R1	ANL A, R1	XRL A, R1	MOV R1, #data [2B]
A	INC R2	DEC R2	ADD A, R2	ADDC A, R2	ORL A, R2	ANL A, R2	XRL A, R2	MOV R2, #data [2B]
B	INC R3	DEC R3	ADD A, R3	ADDC A, R3	ORL A, R3	ANL A, R3	XRL A, R3	MOV R3, #data [2B]
C	INC R4	DEC R4	ADD A, R4	ADDC A, R4	ORL A, R4	ANL A, R4	XRL A, R4	MOV R4, #data [2B]
D	INC R5	DEC R5	ADD A, R5	ADDC A, R5	ORL A, R5	ANL A, R5	XRL A, R5	MOV R5, #data [2B]
E	INC R6	DEC R6	ADD A, R6	ADDC A, R6	ORL A, R6	ANL A, R6	XRL A, R6	MOV R6, #data [2B]
F	INC R7	DEC R7	ADD A, R7	ADDC A, R7	ORL A, R7	ANL A, R7	XRL A, R7	MOV R7, #data [2B]

Note: Key: [2B] = 2 Byte, [3B] = 3 Byte, [2C] = 2 Cycle, [4C] = 4 Cycle, Blank = 1 byte/1 cycle

Instruction Set Summary (Continued)

	8	9	A	B	C	D	E	F
0	SJMP REL [2B, 2C]	MOV DPTR,# data 16 [3B, 2C]	ORL C, /bit [2B, 2C]	ANL C, /bit [2B, 2C]	PUSH dir [2B, 2C]	POP dir [2B, 2C]	MOVX A, @DPTR [2C]	MOVX @DPTR, A [2C]
1	AJMP (P4) [2B, 2C]	ACALL (P4) [2B, 2C]	AJMP (P5) [2B, 2C]	ACALL (P5) [2B, 2C]	AJMP (P6) [2B, 2C]	ACALL (P6) [2B, 2C]	AJMP (P7) [2B, 2C]	ACALL (P7) [2B, 2C]
2	ANL C, bit [2B, 2C]	MOV bit, C [2B, 2C]	MOV C, bit [2B]	CPL bit [2B]	CLR bit [2B]	SETB bit [2B]	MOVX A, @R0 [2C]	MOVX wR0, A [2C]
3	MOVC A, @A + PC [2C]	MOVC A, @A + DPTR [2C]	INC DPTR [2C]	CPL C	CLR C	SETB C	MOVX A, @RI [2C]	MOVX @RI, A [2C]
4	DIV AB [2B, 4C]	SUBB A, #data [2B]	MUL AB [4C]	CJNE A, #data, rel [3B, 2C]	SWAP A	DA A	CLR A	CPL A
5	MOV dir, dir [3B, 2C]	SUBB A, dir [2B]		CJNE A, dir, rel [3B, 2C]	XCH A, dir [2B]	DJNZ dir, rel [3B, 2C]	MOV A, dir [2B]	MOV dir, A [2B]
6	MOV dir, @R0 [2B, 2C]	SUBB A, @R0	MOV @R0, dir [2B, 2C]	CJNE @R0, #data, rel [3B, 2C]	XCH A, @R0	XCHD A, @R0	MOV A, @R0	MOV @R0, A
7	MOV dir, @R1 [2B, 2C]	SUBB A, @R1	MOV @R1, dir [2B, 2C]	CJNE @R1, #data, rel [3B, 2C]	XCH A, @R1	XCHD A, @R1	MOV A, @R1	MOV @R1, A
8	MOV dir, R0 [2B, 2C]	SUBB A, R0	MOV R0, dir [2B, 2C]	CJNE R0, #data, rel [3B, 2C]	XCH A, R0	DJNZ R0, rel [2B, 2C]	MOV A, R0	MOV R0, A
9	MOV dir, R1 [2B, 2C]	SUBB A, R1	MOV R1, dir [2B, 2C]	CJNE R1, #data, rel [3B, 2C]	XCH A, R1	DJNZ R1, rel [2B, 2C]	MOV A, R1	MOV R1, A
A	MOV dir, R2 [2B, 2C]	SUBB A, R2	MOV R2, dir [2B, 2C]	CJNE R2, #data, rel [3B, 2C]	XCH A, R2	DJNZ R2, rel [2B, 2C]	MOV A, R2	MOV R2, A
B	MOV dir, R3 [2B, 2C]	SUBB A, R3	MOV R3, dir [2B, 2C]	CJNE R3, #data, rel [3B, 2C]	XCH A, R3	DJNZ R3, rel [2B, 2C]	MOV A, R3	MOV R3, A
C	MOV dir, R4 [2B, 2C]	SUBB A, R4	MOV R4, dir [2B, 2C]	CJNE R4, #data, rel [3B, 2C]	XCH A, R4	DJNZ R4, rel [2B, 2C]	MOV A, R4	MOV R4, A
D	MOV dir, R5 [2B, 2C]	SUBB A, R5	MOV R5, dir [2B, 2C]	CJNE R5, #data, rel [3B, 2C]	XCH A, R5	DJNZ R5, rel [2B, 2C]	MOV A, R5	MOV R5, A
E	MOV dir, R6 [2B, 2C]	SUBB A, R6	MOV R6, dir [2B, 2C]	CJNE R6, #data, rel [3B, 2C]	XCH A, R6	DJNZ R6, rel [2B, 2C]	MOV A, R6	MOV R6. A
F	MOV dir, R7 [2B, 2C]	SUBB A, R7	MOV R7, dir [2B, 2C]	CJNE R7, #data, rel [3B, 2C]	XCH A, R7	DJNZ R7, rel [2B, 2C]	MOV A, R7	MOV R7, A

Note: Key: [2B] = 2 Byte, [3B] = 3 Byte, [2C] = 2 Cycle, [4C] = 4 Cycle, Blank = 1 byte/1 cycle

Table 1. AT89 Instruction Set Summary[1]

Mnemonic		Description	Byte	Oscillator Period
ARITHMETIC OPERATIONS				
ADD	A,R_n	Add register to Accumulator	1	12
ADD	A,direct	Add direct byte to Accumulator	2	12
ADD	A,@R_i	Add indirect RAM to Accumulator	1	12
ADD	A,#data	Add immediate data to Accumulator	2	12
ADDC	A,R_n	Add register to Accumulator with Carry	1	12
ADDC	A,direct	Add direct byte to Accumulator with Carry	2	12
ADDC	A,@R_i	Add indirect RAM to Accumulator with Carry	1	12
ADDC	A,#data	Add immediate data to Acc with Carry	2	12
SUBB	A,R_n	Subtract Register from Acc with borrow	1	12
SUBB	A,direct	Subtract direct byte from Acc with borrow	2	12
SUBB	A,@R_i	Subtract indirect RAM from ACC with borrow	1	12
SUBB	A,#data	Subtract immediate data from Acc with borrow	2	12
INC	A	Increment Accumulator	1	12
INC	R_n	Increment register	1	12
INC	direct	Increment direct byte	2	12
INC	@R_i	Increment direct RAM	1	12
DEC	A	Decrement Accumulator	1	12
DEC	R_n	Decrement Register	1	12
DEC	direct	Decrement direct byte	2	12
DEC	@R_i	Decrement indirect RAM	1	12
INC	DPTR	Increment Data Pointer	1	24
MUL	AB	Multiply A & B	1	48
DIV	AB	Divide A by B	1	48
DA	A	Decimal Adjust Accumulator	1	12

Mnemonic		Description	Byte	Oscillator Period
LOGICAL OPERATIONS				
ANL	A,R_n	AND Register to Accumulator	1	12
ANL	A,direct	AND direct byte to Accumulator	2	12
ANL	A,@R_i	AND indirect RAM to Accumulator	1	12
ANL	A,#data	AND immediate data to Accumulator	2	12
ANL	direct,A	AND Accumulator to direct byte	2	12
ANL	direct,#data	AND immediate data to direct byte	3	24
ORL	A,R_n	OR register to Accumulator	1	12
ORL	A,direct	OR direct byte to Accumulator	2	12
ORL	A,@R_i	OR indirect RAM to Accumulator	1	12
ORL	A,#data	OR immediate data to Accumulator	2	12
ORL	direct,A	OR Accumulator to direct byte	2	12
ORL	direct,#data	OR immediate data to direct byte	3	24
XRL	A,R_n	Exclusive-OR register to Accumulator	1	12
XRL	A,direct	Exclusive-OR direct byte to Accumulator	2	12
XRL	A,@R_i	Exclusive-OR indirect RAM to Accumulator	1	12
XRL	A,#data	Exclusive-OR immediate data to Accumulator	2	12
XRL	direct,A	Exclusive-OR Accumulator to direct byte	2	12
XRL	direct,#data	Exclusive-OR immediate data to direct byte	3	24
CLR	A	Clear Accumulator	1	12
CPL	A	Complement Accumulator	1	12
RL	A	Rotate Accumulator Left	1	12
RLC	A	Rotate Accumulator Left through the Carry	1	12
LOGICAL OPERATIONS (continued)				

Note: 1. All mnemonics copyrighted © Intel Corp., 1980.

Mnemonic		Description	Byte	Oscillator Period
RR	A	Rotate Accumulator Right	1	12
RRC	A	Rotate Accumulator Right through the Carry	1	12
SWAP	A	Swap nibbles within the Accumulator	1	12
DATA TRANSFER				
MOV	A,R_n	Move register to Accumulator	1	12
MOV	A,direct	Move direct byte to Accumulator	2	12
MOV	A,@R_i	Move indirect RAM to Accumulator	1	12
MOV	A,#data	Move immediate data to Accumulator	2	12
MOV	R_n,A	Move Accumulator to register	1	12
MOV	R_n,direct	Move direct byte to register	2	24
MOV	R_n,#data	Move immediate data to register	2	12
MOV	direct,A	Move Accumulator to direct byte	2	12
MOV	direct,R_n	Move register to direct byte	2	24
MOV	direct,direct	Move direct byte to direct	3	24
MOV	direct,@R_i	Move indirect RAM to direct byte	2	24
MOV	direct,#data	Move immediate data to direct byte	3	24
MOV	@R_i,A	Move Accumulator to indirect RAM	1	12
MOV	@R_i,direct	Move direct byte to indirect RAM	2	24
MOV	@R_i,#data	Move immediate data to indirect RAM	2	12
MOV	DPTR,#data16	Load Data Pointer with a 16-bit constant	3	24
MOVC	A,@A+DPTR	Move Code byte relative to DPTR to Acc	1	24
MOVC	A,@A+PC	Move Code byte relative to PC to Acc	1	24
MOVX	A,@R_i	Move External RAM (8-bit addr) to Acc	1	24
DATA TRANSFER (continued)				

Mnemonic		Description	Byte	Oscillator Period
MOVX	A,@DPTR	Move Exarnal RAM (16-bit addr) to Acc	1	24
MOVX	@R_i,A	Move Acc to External RAM (8-bit addr)	1	24
MOVX	@DPTR,A	Move Acc to External RAM (16-bit addr)	1	24
PUSH	direct	Push direct byte onto stack	2	24
POP	direct	Pop direct byte from stack	2	24
XCH	A,R_n	Exchange register with Accumulator	1	12
XCH	A,direct	Exchange direct byte with Accumulator	2	12
XCH	A,@R_i	Exchange indirect RAM with Accumulator	1	12
XCHD	A,@R_i	Exchange low-order Digit indirect RAM with Acc	1	12
BOOLEAN VARIABLE MANIPULATION				
CLR	C	Clear Carry	1	12
CLR	bit	Clear direct bit	2	12
SETB	C	Set Carry	1	12
SETB	bit	Set direct bit	2	12
CPL	C	Complement Carry	1	12
CPL	bit	Complement direct bit	2	12
ANL	C,bit	AND direct bit to CARRY	2	24
ANL	C,/bit	AND complement of direct bit to Carry	2	24
ORL	C,bit	OR direct bit to Carry	2	24
ORL	C,/bit	OR complement of direct bit to Carry	2	24
MOV	C,bit	Move direct bit to Carry	2	12
MOV	bit,C	Move Carry to direct bit	2	24
JC	rel	Jump if Carry is set	2	24
JNC	rel	Jump if Carry not set	2	24
JB	bit,rel	Jump if direct Bit is set	3	24
JNB	bit,rel	Jump if direct Bit is Not set	3	24
JBC	bit,rel	Jump if direct Bit is set & clear bit	3	24
PROGRAM BRANCHING				

Mnemonic		Description	Byte	Oscillator Period
ACALL	addr11	Absolute Subroutine Call	2	24
LCALL	addr16	Long Subroutine Call	3	24
RET		Return from Subroutine	1	24
RETI		Return from interrupt	1	24
AJMP	addr11	Absolute Jump	2	24
LJMP	addr16	Long Jump	3	24
SJMP	rel	Short Jump (relative addr)	2	24
JMP	@A+DPTR	Jump indirect relative to the DPTR	1	24
JZ	rel	Jump if Accumulator is Zero	2	24
JNZ	rel	Jump if Accumulator is Not Zero	2	24
CJNE	A,direct,rel	Compare direct byte to Acc and Jump if Not Equal	3	24
CJNE	A,#data,rel	Compare immediate to Acc and Jump if Not Equal	3	24
CJNE	R_n,#data,rel	Compare immediate to register and Jump if Not Equal	3	24
CJNE	@R_i,#data,rel	Compare immediate to indirect and Jump if Not Equal	3	24
DJNZ	R_n,rel	Decrement register and Jump if Not Zero	2	24
DJNZ	direct,rel	Decrement direct byte and Jump if Not Zero	3	24
NOP		No Operation	1	12

Table 2. Instruction Opcodes in Hexadecimal Order

Hex Code	Number of Bytes	Mnemonic	Operands
00	1	NOP	
01	2	AJMP	code addr
02	3	LJMP	code addr
03	1	RR	A
04	1	INC	A
05	2	INC	data addr
06	1	INC	@R0
07	1	INC	@R1
08	1	INC	R0
09	1	INC	R1
0A	1	INC	R2
0B	1	INC	R3
0C	1	INC	R4
0D	1	INC	R5
0E	1	INC	R6
0F	1	INC	R7
10	3	JBC	bit addr,code addr
11	2	ACALL	code addr
12	3	LCALL	code addr
13	1	RRC	A
14	1	DEC	A
15	2	DEC	data addr
16	1	DEC	@R0
17	1	DEC	@R1
18	1	DEC	R0
19	1	DEC	R1
1A	1	DEC	R2
1B	1	DEC	R3
1C	1	DEC	R4
1D	1	DEC	R5
1E	1	DEC	R6
1F	1	DEC	R7
20	3	JB	bit addr,code addr
21	2	AJMP	code addr
22	1	RET	
23	1	RL	A
24	2	ADD	A,#data
25	2	ADD	A,data addr

Hex Code	Number of Bytes	Mnemonic	Operands
26	1	ADD	A,@R0
27	1	ADD	A,@R1
28	1	ADD	A,R0
29	1	ADD	A,R1
2A	1	ADD	A,R2
2B	1	ADD	A,R3
2C	1	ADD	A,R4
2D	1	ADD	A,R5
2E	1	ADD	A,R6
2F	1	ADD	A,R7
30	3	JNB	bit addr,code addr
31	2	ACALL	code addr
32	1	RETI	
33	1	RLC	A
34	2	ADDC	A,#data
35	2	ADDC	A,data addr
36	1	ADDC	A,@R0
37	1	ADDC	A,@R1
38	1	ADDC	A,R0
39	1	ADDC	A,R1
3A	1	ADDC	A,R2
3B	1	ADDC	A,R3
3C	1	ADDC	A,R4
3D	1	ADDC	A,R5
3E	1	ADDC	A,R6
3F	1	ADDC	A,R7
40	2	JC	code addr
41	2	AJMP	code addr
42	2	ORL	data addr,A
43	3	ORL	data addr,#data
44	2	ORL	A,#data
45	2	ORL	A,data addr
46	1	ORL	A,@R0
47	1	ORL	A,@R1
48	1	ORL	A,R0
49	1	ORL	A,R1
4A	1	ORL	A,R2

Hex Code	Number of Bytes	Mnemonic	Operands
4B	1	ORL	A,R3
4C	1	ORL	A,R4
4D	1	ORL	A,R5
4E	1	ORL	A,R6
4F	1	ORL	A,R7
50	2	JNC	code addr
51	2	ACALL	code addr
52	2	ANL	data addr,A
53	3	ANL	data addr,#data
54	2	ANL	A,#data
55	2	ANL	A,data addr
56	1	ANL	A,@R0
57	1	ANL	A,@R1
58	1	ANL	A,R0
59	1	ANL	A,R1
5A	1	ANL	A,R2
5B	1	ANL	A,R3
5C	1	ANL	A,R4
5D	1	ANL	A,R5
5E	1	ANL	A,R6
5F	1	ANL	A,R7
60	2	JZ	code addr
61	2	AJMP	code addr
62	2	XRL	data addr,A
63	3	XRL	data addr,#data
64	2	XRL	A,#data
65	2	XRL	A,data addr
66	1	XRL	A,@R0
67	1	XRL	A,@R1
68	1	XRL	A,R0
69	1	XRL	A,R1
6A	1	XRL	A,R2
6B	1	XRL	A,R3
6C	1	XRL	A,R4
6D	1	XRL	A,R5
6E	1	XRL	A,R6
6F	1	XRL	A,R7
70	2	JNZ	code addr

Hex Code	Number of Bytes	Mnemonic	Operands
71	2	ACALL	code addr
72	2	ORL	C,bit addr
73	1	JMP	@A+DPTR
74	2	MOV	A,#data
75	3	MOV	data addr,#data
76	2	MOV	@R0,#data
77	2	MOV	@R1,#data
78	2	MOV	R0,#data
79	2	MOV	R1,#data
7A	2	MOV	R2,#data
7B	2	MOV	R3,#data
7C	2	MOV	R4,#data
7D	2	MOV	R5,#data
7E	2	MOV	R6,#data
7F	2	MOV	R7,#data
80	2	SJMP	code addr
81	2	AJMP	code addr
82	2	ANL	C,bit addr
83	1	MOVC	A,@A+PC
84	1	DIV	AB
85	3	MOV	data addr,data addr
86	2	MOV	data addr,@R0
87	2	MOV	data addr,@R1
88	2	MOV	data addr,R0
89	2	MOV	data addr,R1
8A	2	MOV	data addr,R2
8B	2	MOV	data addr,R3
8C	2	MOV	data addr,R4
8D	2	MOV	data addr,R5
8E	2	MOV	data addr,R6
8F	2	MOV	data addr,R7
90	3	MOV	DPTR,#data
91	2	ACALL	code addr
92	2	MOV	bit addr,C
93	1	MOVC	A,@A+DPTR
94	2	SUBB	A,#data
95	2	SUBB	A,data addr
96	1	SUBB	A,@R0

Hex Code	Number of Bytes	Mnemonic	Operands
97	1	SUBB	A,@R1
98	1	SUBB	A,R0
99	1	SUBB	A,R1
9A	1	SUBB	A,R2
9B	1	SUBB	A,R3
9C	1	SUBB	A,R4
9D	1	SUBB	A,R5
9E	1	SUBB	A,R6
9F	1	SUBB	A,R7
A0	2	ORL	C,/bit addr
A1	2	AJMP	code addr
A2	2	MOV	C,bit addr
A3	1	INC	DPTR
A4	1	MUL	AB
A5		reserved	
A6	2	MOV	@R0,data addr
A7	2	MOV	@R1,data addr
A8	2	MOV	R0,data addr
A9	2	MOV	R1,data addr
AA	2	MOV	R2,data addr
AB	2	MOV	R3,data addr
AC	2	MOV	R4,data addr
AD	2	MOV	R5,data addr
AE	2	MOV	R6,data addr
AF	2	MOV	R7,data addr
B0	2	ANL	C,/bit addr
B1	2	ACALL	code addr
B2	2	CPL	bit addr
B3	1	CPL	C
B4	3	CJNE	A,#data,code addr
B5	3	CJNE	A,data addr,code addr
B6	3	CJNE	@R0,#data,code addr
B7	3	CJNE	@R1,#data,code addr
B8	3	CJNE	R0,#data,code addr
B9	3	CJNE	R1,#data,code addr
BA	3	CJNE	R2,#data,code addr
BB	3	CJNE	R3,#data,code addr
BC	3	CJNE	R4,#data,code addr

Hex Code	Number of Bytes	Mnemonic	Operands
BD	3	CJNE	R5,#data,code addr
BE	3	CJNE	R6,#data,code addr
BF	3	CJNE	R7,#data,code addr
C0	2	PUSH	data addr
C1	2	AJMP	code addr
C2	2	CLR	bit addr
C3	1	CLR	C
C4	1	SWAP	A
C5	2	XCH	A,data addr
C6	1	XCH	A,@R0
C7	1	XCH	A,@R1
C8	1	XCH	A,R0
C9	1	XCH	A,R1
CA	1	XCH	A,R2
CB	1	XCH	A,R3
CC	1	XCH	A,R4
CD	1	XCH	A,R5
CE	1	XCH	A,R6
CF	1	XCH	A,R7
D0	2	POP	data addr
D1	2	ACALL	code addr
D2	2	SETB	bit addr
D3	1	SETB	C
D4	1	DA	A
D5	3	DJNZ	data addr,code addr
D6	1	XCHD	A,@R0
D7	1	XCHD	A,@R1
D8	2	DJNZ	R0,code addr
D9	2	DJNZ	R1,code addr
DA	2	DJNZ	R2,code addr
DB	2	DJNZ	R3,code addr
DC	2	DJNZ	R4,code addr
DD	2	DJNZ	R5,code addr
DE	2	DJNZ	R6,code addr
DF	2	DJNZ	R7,code addr
E0	1	MOVX	A,@DPTR
E1	2	AJMP	code addr
E2	1	MOVX	A,@R0

Hex Code	Number of Bytes	Mnemonic	Operands
E3	1	MOVX	A,@R1
E4	1	CLR	A
E5	2	MOV	A,data addr
E6	1	MOV	A,@R0
E7	1	MOV	A,@R1
E8	1	MOV	A,R0
E9	1	MOV	A,R1
EA	1	MOV	A,R2
EB	1	MOV	A,R3
EC	1	MOV	A,R4
ED	1	MOV	A,R5
EE	1	MOV	A,R6
EF	1	MOV	A,R7
F0	1	MOVX	@DPTR,A
F1	2	ACALL	code addr
F2	1	MOVX	@R0,A
F3	1	MOVX	@R1,A
F4	1	CPL	A
F5	2	MOV	data addr,A
F6	1	MOV	@R0,A
F7	1	MOV	@R1,A
F8	1	MOV	R0,A
F9	1	MOV	R1,A
FA	1	MOV	R2,A
FB	1	MOV	R3,A
FC	1	MOV	R4,A
FD	1	MOV	R5,A
FE	1	MOV	R6,A
FF	1	MOV	R7,A

Instruction Definitions

ACALL addr11

Function: Absolute Call

Description: ACALL unconditionally calls a subroutine located at the indicated address. The instruction increments the PC twice to obtain the address of the following instruction, then pushes the 16-bit result onto the stack (low-order byte first) and increments the Stack Pointer twice. The destination address is obtained by successively concatenating the five high-order bits of the incremented PC, opcode bits 7 through 5, and the second byte of the instruction. The subroutine called must therefore start within the same 2 K block of the program memory as the first byte of the instruction following ACALL. No flags are affected.

Example: Initially SP equals 07H. The label SUBRTN is at program memory location 0345 H. After executing the following instruction,

```
ACALL      SUBRTN
```

at location 0123H, SP contains 09H, internal RAM locations 08H and 09H will contain 25H and 01H, respectively, and the PC contains 0345H.

Bytes: 2

Cycles: 2

Encoding:

a10	a9	a8	1	0	0	0	1	a7	a6	a5	a4	a3	a2	a1	a0

Operation: ACALL
(PC) ← (PC) + 2
(SP) ← (SP) + 1
((SP)) ← (PC_{7-0})
(SP) ← (SP) + 1
((SP)) ← (PC_{15-8})
(PC_{10-0}) ← page address

ADD A,<src-byte>

Function: Add

Description: ADD adds the byte variable indicated to the Accumulator, leaving the result in the Accumulator. The carry and auxiliary-carry flags are set, respectively, if there is a carry-out from bit 7 or bit 3, and cleared otherwise. When adding unsigned integers, the carry flag indicates an overflow occurred.

OV is set if there is a carry-out of bit 6 but not out of bit 7, or a carry-out of bit 7 but not bit 6; otherwise, OV is cleared. When adding signed integers, OV indicates a negative number produced as the sum of two positive operands, or a positive sum from two negative operands.

Four source operand addressing modes are allowed: register, direct, register-indirect, or immediate.

Example: The Accumulator holds 0C3H (11000011B), and register 0 holds 0AAH (10101010B). The following instruction,

```
ADD         A,R0
```

leaves 6DH (01101101B) in the Accumulator with the AC flag cleared and both the carry flag and OV set to 1.

ADD A,R_n

Bytes: 1

Cycles: 1

Encoding:

0	0	1	0	1	r	r	r

Operation: ADD
(A) ← (A) + (R_n)

ADD A,direct

Bytes: 2

Cycles: 1

Encoding:

0	0	1	0	0	1	0	1	direct address

Operation: ADD
(A) ← (A) + (direct)

ADD A,@R_i

Bytes: 1

Cycles: 1

Encoding:

0	0	1	0	0	1	1	i

Operation: ADD
(A) ← (A) + ((Ri))

ADD A,#data

Bytes: 2

Cycles: 1

Encoding:

0	0	1	0	0	1	0	0	immediate data

Operation: ADD
(A) ← (A) + #data

ADDC A, <src-byte>

Function: Add with Carry

Description: ADDC simultaneously adds the byte variable indicated, the carry flag and the Accumulator contents, leaving the result in the Accumulator. The carry and auxiliary-carry flags are set respectively, if there is a carry-out from bit 7 or bit 3, and cleared otherwise. When adding unsigned integers, the carry flag indicates an overflow occurred.

OV is set if there is a carry-out of bit 6 but not out of bit 7, or a carry-out of bit 7 but not out of bit 6; otherwise OV is cleared. When adding signed integers, OV indicates a negative number produced as the sum of two positive operands or a positive sum from two negative operands.

Four source operand addressing modes are allowed: register, direct, register-indirect, or immediate.

Example: The Accumulator holds 0C3H (11000011B) and register 0 holds 0AAH (10101010B) with the carry flag set. The following instruction,

```
ADDC        A,R0
```

leaves 6EH (01101110B) in the Accumulator with AC cleared and both the Carry flag and OV set to 1.

ADDC A,R_n

Bytes: 1

Cycles: 1

Encoding:

0	0	1	1	1	r	r	r

Operation: ADDC
(A) ← (A) + (C) + (R_n)

ADDC A,direct

Bytes: 2

Cycles: 1

Encoding:

0	0	1	1	0	1	0	1	direct address

Operation: ADDC
(A) ← (A) + (C) + (direct)

ADDC A,@R_i

Bytes: 1

Cycles: 1

Encoding:

0	0	1	1	0	1	1	i

Operation: ADDC
(A) ← (A) + (C) + ((Ri))

ADDC A,#data

Bytes: 2

Cycles: 1

Encoding:

0	0	1	1	0	1	0	0	immediate data

Operation: ADDC
(A) ← (A) + (C) + #data

AJMP addr11

Function: Absolute Jump

Description: AJMP transfers program execution to the indicated address, which is formed at run-time by concatenating the high-order five bits of the PC (after incrementing the PC twice), opcode bits 7 through 5, and the second byte of the instruction. The destination must therfore be within the same 2 K block of program memory as the first byte of the instruction following AJMP.

Example: The label JMPADR is at program memory location 0123H. The following instruction,

AJMP JMPADR

is at location 0345H and loads the PC with 0123H.

Bytes: 2

Cycles: 2

Encoding:

a10	a9	a8	0	0	0	0	1	a7	a6	a5	a4	a3	a2	a1	a0

Operation: AJMP
(PC) ← (PC) + 2
$(PC_{10\text{-}0})$ ← page address

ANL <dest-byte>,<src-byte>

Function: Logical-AND for byte variables

Description: ANL performs the bitwise logical-AND operation between the variables indicated and stores the results in the destination variable. No flags are affected.

The two operands allow six addressing mode combinations. When the destination is the Accumulator, the source can use register, direct, register-indirect, or immediate addressing; when the destination is a direct address, the source can be the Accumulator or immediate data.

Note: When this instruction is used to modify an output port, the value used as the original port data will be read from the output data latch, *not* the input pins.

Example: If the Accumulator holds 0C3H (1100001lB), and register 0 holds 55H (01010101B), then the following instruction,

ANL A,R0

leaves 41H (01000001B) in the Accumulator.

When the destination is a directly addressed byte, this instruction clears combinations of bits in any RAM location or hardware register. The mask byte determining the pattern of bits to be cleared would either be a constant contained in the instruction or a value computed in the Accumulator at run-time. The following instruction,

ANL P1,#01110011B

clears bits 7, 3, and 2 of output port 1.

ANL A,R_n

Bytes: 1

Cycles: 1

Encoding:

0	1	0	1	1	r	r	r

Operation: ANL
(A) ← (A) ∧ (R_n)

ANL A,direct

Bytes: 2

Cycles: 1

Encoding:

0	1	0	1	0	1	0	1	direct address

Operation: ANL
(A) ← (A) ∧ (direct)

ANL A,@R_i

Bytes: 1

Cycles: 1

Encoding:

0	1	0	1	0	1	1	i

Operation: ANL
(A) ← (A) ∧ ((R_i))

ANL A,#data

Bytes: 2

Cycles: 1

Encoding:

0	1	0	1	0	1	0	0	immediate data

Operation: ANL
(A) ← (A) ∧ #data

ANL direct,A

Bytes: 2

Cycles: 1

Encoding:

0	1	0	1	0	0	1	0	direct address

Operation: ANL
(direct) ← (direct) ∧ (A)

ANL direct,#data

Bytes: 3

Cycles: 2

Encoding:

0	1	0	1	0	0	1	1	direct address	immediate data

Operation: ANL
(direct) ← (direct) ∧ #data

ANL C,<src-bit>

Function: Logical-AND for bit variables

Description: If the Boolean value of the source bit is a logical 0, then ANL C clears the carry flag; otherwise, this instruction leaves the carry flag in its current state. A slash (/) preceding the operand in the assembly language indicates that the logical complement of the addressed bit is used as the source value, *but the source bit itself is not affected*. No other flags are affected.

Only direct addressing is allowed for the source operand.

Example: Set the carry flag if, and only if, P1.0 = 1, ACC.7 = 1, and OV = 0:

```
MOV     C,P1.0      ;LOAD CARRY WITH INPUT PIN STATE
ANL     C,ACC.7     ;AND CARRY WITH ACCUM. BIT 7
ANL     C,/OV       ;AND WITH INVERSE OF OVERFLOW FLAG
```

ANL C,bit

Bytes: 2

Cycles: 2

Encoding:

1	0	0	0	0	0	1	0	bit address

Operation: ANL
(C) ← (C) ∧ (bit)

ANL C,/bit

Bytes: 2

Cycles: 2

Encoding:

1	0	1	1	0	0	0	0	bit address

Operation: ANL
(C) ← (C) ∧ ¬ (bit)

CJNE <dest-byte>,<src-byte>, rel

Function: Compare and Jump if Not Equal.

Description: CJNE compares the magnitudes of the first two operands and branches if their values are not equal. The branch destination is computed by adding the signed relative-displacement in the last instruction byte to the PC, after incrementing the PC to the start of the next instruction. The carry flag is set if the unsigned integer value of <dest-byte> is less than the unsigned integer value of <src-byte>; otherwise, the carry is cleared. Neither operand is affected.

The first two operands allow four addressing mode combinations: the Accumulator may be compared with any directly addressed byte or immediate data, and any indirect RAM location or working register can be compared with an immediate constant.

Example: The Accumulator contains 34H. Register 7 contains 56H. The first instruction in the sequence,

```
                CJNE    R7, # 60H, NOT_EQ
;               . . .   . . . . .       ;R7 = 60H.
NOT_EQ:         JC      REQ_LOW         ;IF R7 < 60H.
;               . . .   . . . . .       ;R7 > 60H.
```

sets the carry flag and branches to the instruction at label NOT_EQ. By testing the carry flag, this instruction determines whether R7 is greater or less than 60H.

If the data being presented to Port 1 is also 34H, then the following instruction,

```
WAIT:   CJNE    A, P1,WAIT
```

clears the carry flag and continues with the next instruction in sequence, since the Accumulator does equal the data read from P1. (If some other value was being input on P1, the program loops at this point until the P1 data changes to 34H.)

CJNE A,direct,rel

Bytes: 3

Cycles: 2

Encoding:

1	0	1	1	0	1	0	1	direct address	rel. address

Operation:
(PC) ← (PC) + 3
IF (A) < > (*direct*)
THEN
    (PC) ← (PC) + *relative offset*
IF (A) < (*direct*)
THEN
    (C) ← 1
ELSE
    (C) ← 0

CJNE A,#data,rel

Bytes: 3

Cycles: 2

Encoding:

1	0	1	1	0	1	0	0	immediate data	rel. address

Operation:
(PC) ← (PC) + 3
IF (A) < > *data*
THEN
    (PC) ← (PC) + *relative offset*
IF (A) < *data*
THEN
    (C) ← 1
ELSE
    (C) ← 0

CJNE R_n,#data,rel

Bytes: 3

Cycles: 2

Encoding:

1	0	1	1	1	r	r	r	immediate data	rel. address

Operation:
(PC) ← (PC) + 3
IF (R_n) < > *data*
THEN
    (PC) ← (PC) + *relative offset*
IF (R_n) < *data*
THEN
    (C) ← 1
ELSE
    (C) ← 0

CJNE @R_i,data,rel

Bytes: 3

Cycles: 2

Encoding:

1	0	1	1	0	1	1	i	immediate data	rel. address

Operation:
(PC) ← (PC) + 3
IF ((R$_i$)) < > *data*
THEN
    (PC) ← (PC) + *relative offset*
IF ((R$_i$)) < *data*
THEN
    (C) ← 1
ELSE
    (C) ← 0

CLR A

Function: Clear Accumulator

Description: CLR A clears the Accumulator (all bits set to 0). No flags are affected

Example: The Accumulator contains 5CH (01011100B). The following instruction,CLR Aleaves the Accumulator set to 00H (00000000B).

Bytes: 1

Cycles: 1

Encoding:

1	1	1	0	0	1	0	0

Operation:
CLR
(A) ← 0

CLR bit

Function: Clear bit

Description: CLR bit clears the indicated bit (reset to 0). No other flags are affected. CLR can operate on the carry flag or any directly addressable bit.

Example: Port 1 has previously been written with 5DH (01011101B). The following instruction,CLR P1.2 leaves the port set to 59H (01011001B).

CLR C

Bytes: 1

Cycles: 1

Encoding:

1	1	0	0	0	0	1	1

Operation:
CLR
(C) ← 0

CLR bit

Bytes: 2

Cycles: 1

Encoding:

1	1	0	0	0	0	1	0	bit address

Operation:
CLR
(bit) ← 0

CPL A

Function: Complement Accumulator

Description: CPLA logically complements each bit of the Accumulator (one's complement). Bits which previously contained a 1 are changed to a 0 and vice-versa. No flags are affected.

Example: The Accumulator contains 5CH (01011100B). The following instruction,

```
CPL     A
```

leaves the Accumulator set to 0A3H (10100011B).

Bytes: 1

Cycles: 1

Encoding:

1	1	1	1	0	1	0	0

Operation: CPL
(A) ← ¬ (A)

CPL bit

Function: Complement bit

Description: CPL bit complements the bit variable specified. A bit that had been a 1 is changed to 0 and vice-versa. No other flags are affected. CLR can operate on the carry or any directly addressable bit.

Note: When this instruction is used to modify an output pin, the value used as the original data is read from the output data latch, *not* the input pin.

Example: Port 1 has previously been written with 5BH (01011101B). The following instruction sequence,CPL P1.1CPL P1.2 leaves the port set to 5BH (01011011B).

CPL C

Bytes: 1

Cycles: 1

Encoding:

1	0	1	1	0	0	1	1

Operation: CPL
(C) ← ¬ (C)

CPL bit

Bytes: 2

Cycles: 1

Encoding:

1	0	1	1	0	0	1	0

bit address

Operation: CPL
(bit) ← ¬ (bit)

DA A

Function: Decimal-adjust Accumulator for Addition

Description: DA A adjusts the eight-bit value in the Accumulator resulting from the earlier addition of two variables (each in packed-BCD format), producing two four-bit digits. Any ADD or ADDC instruction may have been used to perform the addition.

If Accumulator bits 3 through 0 are greater than nine (xxxx1010-xxxx1111), or if the AC flag is one, six is added to the Accumulator producing the proper BCD digit in the low-order nibble. This internal addition sets the carry flag if a carry-out of the low-order four-bit field propagates through all high-order bits, but it does not clear the carry flag otherwise.

If the carry flag is now set, or if the four high-order bits now exceed nine (1010xxxx-1111xxxx), these high-order bits are incremented by six, producing the proper BCD digit in the high-order nibble. Again, this sets the carry flag if there is a carry-out of the high-order bits, but does not clear the carry. The carry flag thus indicates if the sum of the original two BCD variables is greater than 100, allowing multiple precision decimal addition. OV is not affected.

All of this occurs during the one instruction cycle. Essentially, this instruction performs the decimal conversion by adding 00H, 06H, 60H, or 66H to the Accumulator, depending on initial Accumulator and PSW conditions.

Note: DA A *cannot* simply convert a hexadecimal number in the Accumulator to BCD notation, nor does DAA apply to decimal subtraction.

Example: The Accumulator holds the value 56H (01010110B), representing the packed BCD digits of the decimal number 56. Register 3 contains the value 67H (01100111B), representing the packed BCD digits of the decimal number 67. The carry flag is set. The following instruction sequence

```
ADDC    A,R3
DA      A
```

first performs a standard two's-complement binary addition, resulting in the value 0BEH (10111110) in the Accumulator. The carry and auxiliary carry flags are cleared.

The Decimal Adjust instruction then alters the Accumulator to the value 24H (00100100B), indicating the packed BCD digits of the decimal number 24, the low-order two digits of the decimal sum of 56, 67, and the carry-in. The carry flag is set by the Decimal Adjust instruction, indicating that a decimal overflow occurred. The true sum of 56, 67, and 1 is 124.

BCD variables can be incremented or decremented by adding 01H or 99H. If the Accumulator initially holds 30H (representing the digits of 30 decimal), then the following instruction sequence,

```
ADD     A, # 99H
DA      A
```

leaves the carry set and 29H in the Accumulator, since 30 + 99 = 129. The low-order byte of the sum can be interpreted to mean 30 - 1 = 29.

Bytes: 1

Cycles: 1

Encoding:

1	1	0	1	0	1	0	0

Operation: DA
-contents of Accumulator are BCD
IF [[(A_{3-0}) > 9] ∨ [(AC) = 1]]
THEN $(A_{3-0}) \leftarrow (A_{3-0}) + 6$
AND
IF [[(A_{7-4}) > 9] ∨ [(C) = 1]]
THEN $(A_{7-4}) \leftarrow (A_{7-4}) + 6$

DEC byte

Function: Decrement

Description: DEC byte decrements the variable indicated by 1. An original value of 00H underflows to 0FFH. No flags are affected. Four operand addressing modes are allowed: accumulator, register, direct, or register-indirect.

Note: When this instruction is used to modify an output port, the value used as the original port data will be read from the output data latch, *not* the input pins.

Example: Register 0 contains 7FH (01111111B). Internal RAM locations 7EH and 7FH contain 00H and 40H, respectively. The following instruction sequence,

```
DEC     @R0
DEC     R0
DEC     @R0
```

leaves register 0 set to 7EH and internal RAM locations 7EH and 7FH set to 0FFH and 3FH.

DEC A

Bytes: 1

Cycles: 1

Encoding:

0	0	0	1	0	1	0	0

Operation: DEC
$(A) \leftarrow (A) - 1$

DEC R_n

Bytes: 1

Cycles: 1

Encoding:

0	0	0	1	1	r	r	r

Operation: DEC
$(R_n) \leftarrow (R_n) - 1$

DEC direct

Bytes: 2

Cycles: 1

Encoding:

0	0	0	1	0	1	0	1	direct address

Operation: DEC
(direct) ← (direct) - 1

DEC @R_i

Bytes: 1

Cycles: 1

Encoding:

0	0	0	1	0	1	1	i

Operation: DEC
$((R_i)) \leftarrow ((R_i)) - 1$

DIV AB

Function: Divide

Description: DIV AB divides the unsigned eight-bit integer in the Accumulator by the unsigned eight-bit integer in register B. The Accumulator receives the integer part of the quotient; register B receives the integer remainder. The carry and OV flags are cleared.

Exception: if B had originally contained 00H, the values returned in the Accumulator and B-register are undefined and the overflow flag are set. The carry flag is cleared in any case.

Example: The Accumulator contains 251 (0FBH or 11111011B) and B contains 18 (12H or 00010010B). The following instruction,

```
DIV     AB
```

leaves 13 in the Accumulator (0DH or 00001101B) and the value 17 (11H or 00010001B) in B, since 251 = (13 x 18) + 17. Carry and OV are both cleared.

Bytes: 1

Cycles: 4

Encoding:

1	0	0	0	0	1	0	0

Operation: DIV
$(A)_{15-8} \leftarrow (A)/(B)$
$(B)_{7-0}$

DJNZ <byte>,<rel-addr>

Function: Decrement and Jump if Not Zero

Description: DJNZ decrements the location indicated by 1, and branches to the address indicated by the second operand if the resulting value is not zero. An original value of 00H underflows to 0FFH. No flags are affected. The branch destination is computed by adding the signed relative-displacement value in the last instruction byte to the PC, after incrementing the PC to the first byte of the following instruction.

The location decremented may be a register or directly addressed byte.

Note: When this instruction is used to modify an output port, the value used as the original port data will be read from the output data latch, *not* the input pins.

Example: Internal RAM locations 40H, 50H, and 60H contain the values 01H, 70H, and 15H, respectively. The following instruction sequence,

```
DJNZ    40H,LABEL_1
DJNZ    50H,LABEL_2
DJNZ    60H,LABEL_3
```

causes a jump to the instruction at label LABEL_2 with the values 00H, 6FH, and 15H in the three RAM locations. The first jump was *not* taken because the result was zero.

This instruction provides a simple way to execute a program loop a given number of times or for adding a moderate time delay (from 2 to 512 machine cycles) with a single instruction. The following instruction sequence,

```
        MOV     R2, # 8
TOGGLE: CPL     P1.7
        DJNZ    R2,TOGGLE
```

toggles P1.7 eight times, causing four output pulses to appear at bit 7 of output Port 1. Each pulse lasts three machine cycles; two for DJNZ and one to alter the pin.

DJNZ R_n,rel

Bytes: 2

Cycles: 2

Encoding:

1	1	0	1	1	r	r	r	rel. address

Operation: DJNZ
(PC) ← (PC) + 2
(R_n) ← (R_n) - 1
IF (R_n) > 0 or (R_n) < 0
THEN
(PC) ← (PC) + rel

DJNZ direct,rel

Bytes: 3

Cycles: 2

Encoding:

1	1	0	1	0	1	0	1	direct address	rel. address

Operation: DJNZ
(PC) ← (PC) + 2
(direct) ← (direct) - 1
IF (direct) > 0 or (direct) < 0
THEN
(PC) ← (PC) + rel

INC <byte>

Function: Increment

Description: INC increments the indicated variable by 1. An original value of 0FFH overflows to 00H. No flags are affected. Three addressing modes are allowed: register, direct, or register-indirect.

Note: When this instruction is used to modify an output port, the value used as the original port data will be read from the output data latch, *not* the input pins.

Example: Register 0 contains 7EH (011111110B). Internal RAM locations 7EH and 7FH contain 0FFH and 40H, respectively. The following instruction sequence,

```
INC     @R0
INC     R0
INC     @R0
```

leaves register 0 set to 7FH and internal RAM locations 7EH and 7FH holding 00H and 41H, respectively.

INC A

Bytes: 1

Cycles: 1

Encoding:

0	0	0	0	0	1	0	0

Operation: INC
(A) ← (A) + 1

INC R_n

Bytes: 1

Cycles: 1

Encoding:

0	0	0	0	1	r	r	r

Operation: INC
(R_n) ← (R_n) + 1

INC direct

Bytes: 2

Cycles: 1

Encoding:

0	0	0	0	0	1	0	1	direct address

Operation: INC
(direct) ← (direct) + 1

INC @R_i

Bytes: 1

Cycles: 1

Encoding:

0	0	0	0	0	1	1	i

Operation: INC
((R_i)) ← ((R_i)) + 1

INC DPTR

Function: Increment Data Pointer

Description: INC DPTR increments the 16-bit data pointer by 1. A 16-bit increment (modulo 2^{16}) is performed, and an overflow of the low-order byte of the data pointer (DPL) from 0FFH to 00H increments the high-order byte (DPH). No flags are affected.

This is the only 16-bit register which can be incremented.

Example: Registers DPH and DPL contain 12H and 0FEH, respectively. The following instruction sequence,

```
INC     DPTR
INC     DPTR
INC     DPTR
```

changes DPH and DPL to 13H and 01H.

Bytes: 1

Cycles: 2

Encoding:

1	0	1	0	0	0	1	1

Operation: INC
(DPTR) ← (DPTR) + 1

JB bit,rel

Function: Jump if Bit set

Description: If the indicated bit is a one, JB jump to the address indicated; otherwise, it proceeds with the next instruction. The branch destination is computed by adding the signed relative-displacement in the third instruction byte to the PC, after incrementing the PC to the first byte of the next instruction. The bit tested is not modified. No flags are affected.

Example: The data present at input port 1 is 11001010B. The Accumulator holds 56 (01010110B). The following instruction sequence,

```
JB      P1.2,LABEL1
JB      ACC. 2,LABEL2
```

causes program execution to branch to the instruction at label LABEL2.

Bytes: 3

Cycles: 2

Encoding:

0	0	1	0	0	0	0	0	bit address	rel. address

Operation:
```
JB
(PC) ← (PC) + 3
IF   (bit) = 1
     THEN
          (PC) ← (PC) + rel
```


JBC bit,rel

Function: Jump if Bit is set and Clear bit

Description: If the indicated bit is one, JBC branches to the address indicated; otherwise, it proceeds with the next instruction. *The bit will not be cleared if it is already a zero.* The branch destination is computed by adding the signed relative-displacement in the third instruction byte to the PC, after incrementing the PC to the first byte of the next instruction. No flags are affected.

Note: When this instruction is used to test an output pin, the value used as the original data will be read from the output data latch, *not* the input pin.

Example: The Accumulator holds 56H (01010110B). The following instruction sequence,

```
JBC     ACC.3,LABEL1
JBC     ACC.2,LABEL2
```

causes program execution to continue at the instruction identified by the label LABEL2, with the Accumulator modified to 52H (01010010B).

Bytes: 3

Cycles: 2

Encoding:

0	0	0	1	0	0	0	0	bit address	rel. address

Operation:
```
JBC
(PC) ← (PC) + 3
IF   (bit) = 1
     THEN
          (bit) ← 0
          (PC) ← (PC) +rel
```

JC rel

Function: Jump if Carry is set

Description: If the carry flag is set, JC branches to the address indicated; otherwise, it proceeds with the next instruction. The branch destination is computed by adding the signed relative-displacement in the second instruction byte to the PC, after incrementing the PC twice. No flags are affected.

Example: The carry flag is cleared. The following instruction sequence,

```
JC      LABEL1
CPL     C
JC      LABEL 2
```

sets the carry and causes program execution to continue at the instruction identified by the label LABEL2.

Bytes: 2

Cycles: 2

Encoding:

0	1	0	0	0	0	0	0	rel. address

Operation:
```
JC
(PC) ← (PC) + 2
IF   (C) = 1
     THEN
          (PC) ← (PC) + rel
```

JMP @A+DPTR

Function: Jump indirect

Description: JMP @A+DPTR adds the eight-bit unsigned contents of the Accumulator with the 16-bit data pointer and loads the resulting sum to the program counter. This is the address for subsequent instruction fetches. Sixteen-bit addition is performed (modulo 2^{16}): a carry-out from the low-order eight bits propagates through the higher-order bits. Neither the Accumulator nor the Data Pointer is altered. No flags are affected.

Example: An even number from 0 to 6 is in the Accumulator. The following sequence of instructions branches to one of four AJMP instructions in a jump table starting at JMP_TBL.

```
        MOV     DPTR, # JMP_TBL
        JMP     @A + DPTR
JMP_TBL: AJMP   LABEL0
        AJMP    LABEL1
        AJMP    LABEL2
        AJMP    LABEL3
```

If the Accumulator equals 04H when starting this sequence, execution jumps to label LABEL2. Because AJMP is a 2-byte instruction, the jump instructions start at every other address.

Bytes: 1

Cycles: 2

Encoding:

0	1	1	1	0	0	1	1

Operation: JMP
(PC) ← (A) + (DPTR)

JNB bit,rel

Function: Jump if Bit Not set

Description: If the indicated bit is a 0, JNB branches to the indicated address; otherwise, it proceeds with the next instruction. The branch destination is computed by adding the signed relative-displacement in the third instruction byte to the PC, after incrementing the PC to the first byte of the next instruction. *The bit tested is not modified.* No flags are affected.

Example: The data present at input port 1 is 11001010B. The Accumulator holds 56H (01010110B). The following instruction sequence,

```
JNB     P1.3,LABEL1
JNB     ACC.3,LABEL2
```

causes program execution to continue at the instruction at label LABEL2.

Bytes: 3

Cycles: 2

Encoding:

0	0	1	1	0	0	0	0	bit address	rel. address

Operation: JNB
(PC) ← (PC) + 3
IF (bit) = 0
THEN (PC) ← (PC) + rel

JNC rel

Function: Jump if Carry not set

Description: If the carry flag is a 0, JNC branches to the address indicated; otherwise, it proceeds with the next instruction. The branch destination is computed by adding the signal relative-displacement in the second instruction byte to the PC, after incrementing the PC twice to point to the next instruction. The carry flag is not modified.

Example: The carry flag is set. The following instruction sequence,

```
JNC     LABEL1
CPL     C
JNC     LABEL2
```

clears the carry and causes program execution to continue at the instruction identified by the label LABEL2.

Bytes: 2

Cycles: 2

Encoding:

0	1	0	1	0	0	0	0	rel. address

Operation: JNC
(PC) ← (PC) + 2
IF (C) = 0
THEN (PC) ← (PC) + rel

JNZ rel

Function: Jump if Accumulator Not Zero

Description: If any bit of the Accumulator is a one, JNZ branches to the indicated address; otherwise, it proceeds with the next instruction. The branch destination is computed by adding the signed relative-displacement in the second instruction byte to the PC, after incrementing the PC twice. The Accumulator is not modified. No flags are affected.

Example: The Accumulator originally holds 00H. The following instruction sequence,

```
JNZ     LABEL1
INC     A
JNZ     LABEL2
```

sets the Accumulator to 01H and continues at label LABEL2.

Bytes: 2

Cycles: 2

Encoding:

0	1	1	1	0	0	0	0	rel. address

Operation: JNZ
(PC) ← (PC) + 2
IF (A) ≠ 0
THEN (PC) ← (PC) + rel

JZ rel

Function: Jump if Accumulator Zero

Description: If all bits of the Accumulator are 0, JZ branches to the address indicated; otherwise, it proceeds with the next instruction. The branch destination is computed by adding the signed relative-displacement in the second instruction byte to the PC, after incrementing the PC twice. The Accumulator is not modified. No flags are affected.

Example: The Accumulator originally contains 01H. The following instruction sequence,

```
JZ      LABEL1
DEC     A
JZ      LABEL2
```

changes the Accumulator to 00H and causes program execution to continue at the instruction identified by the label LABEL2.

Bytes: 2

Cycles: 2

Encoding:

0	1	1	0	0	0	0	0	rel. address

Operation: JZ
(PC) ← (PC) + 2
IF (A) = 0
THEN (PC) ← (PC) + rel

LCALL addr16

Function: Long call

Description: LCALL calls a subroutine located at the indicated address. The instruction adds three to the program counter to generate the address of the next instruction and then pushes the 16-bit result onto the stack (low byte first), incrementing the Stack Pointer by two. The high-order and low-order bytes of the PC are then loaded, respectively, with the second and third bytes of the LCALL instruction. Program execution continues with the instruction at this address. The subroutine may therefore begin anywhere in the full 64K byte program memory address space. No flags are affected.

Example: Initially the Stack Pointer equals 07H. The label SUBRTN is assigned to program memory location 1234H. After executing the instruction,

```
LCALL   SUBRTN
```

at location 0123H, the Stack Pointer will contain 09H, internal RAM locations 08H and 09H will contain 26H and 01H, and the PC will contain 1234H.

Bytes: 3

Cycles: 2

Encoding:

0	0	0	1	0	0	1	0	addr15-addr8	addr7-addr0

Operation: LCALL
(PC) ← (PC) + 3
(SP) ← (SP) + 1
((SP)) ← (PC_{7-0})
(SP) ← (SP) + 1
((SP)) ← (PC_{15-8})
(PC) ← $addr_{15-0}$

LJMP addr16

Function: Long Jump

Description: LJMP causes an unconditional branch to the indicated address, by loading the high-order and low-order bytes of the PC (respectively) with the second and third instruction bytes. The destination may therefore be anywhere in the full 64K program memory address space. No flags are affected.

Example: The label JMPADR is assigned to the instruction at program memory location 1234H. The instruction,

```
LJMP    JMPADR
```

at location 0123H will load the program counter with 1234H.

Bytes: 3

Cycles: 2

Encoding:

0	0	0	0	0	0	1	0	addr15-addr8	addr7-addr0

Operation: LJMP
(PC) ← $addr_{15-0}$

MOV <dest-byte>,<src-byte>

Function: Move byte variable

Description: The byte variable indicated by the second operand is copied into the location specified by the first operand. The source byte is not affected. No other register or flag is affected.

This is by far the most flexible operation. Fifteen combinations of source and destination addressing modes are allowed.

Example: Internal RAM location 30H holds 40H. The value of RAM location 40H is 10H. The data present at input port 1 is 11001010B (0CAH).

```
MOV     R0,#30H     ;R0 < = 30H
MOV     A,@R0       ;A < = 40H
MOV     R1,A        ;R1 < = 40H
MOV     B,@R1       ;B < = 10H
MOV     @R1,P1      ;RAM (40H) < = 0CAH
MOV     P2,P1       ;P2 #0CAH
```

leaves the value 30H in register 0, 40H in both the Accumulator and register 1, 10H in register B, and 0CAH (11001010B) both in RAM location 40H and output on port 2.

MOV A,R_n

Bytes: 1

Cycles: 1

Encoding:

1	1	1	0	1	r	r	r

Operation: MOV
(A) ← (R_n)

*MOV A,direct

Bytes: 2

Cycles: 1

Encoding:

1	1	1	0	0	1	0	1	direct address

Operation: MOV
(A) ← (direct)

*** MOV A,ACC is not a valid Instruction.**

MOV A,@R_i

Bytes: 1

Cycles: 1

Encoding:

1	1	1	0	0	1	1	i

Operation: MOV
(A) ← ((Ri))

MOV A,#data

Bytes: 2

Cycles: 1

Encoding:

0	1	1	1	0	1	0	0	immediate data

Operation: MOV
(A) ← #data

MOV R_n,A

Bytes: 1

Cycles: 1

Encoding:

1	1	1	1	1	r	r	r

Operation: MOV
(R_n) ← (A)

MOV R_n,direct

Bytes: 2

Cycles: 2

Encoding:

1	0	1	0	1	r	r	r	direct addr.

Operation: MOV
(R_n) ← (direct)

MOV R_n,#data

Bytes: 2

Cycles: 1

Encoding:

0	1	1	1	1	r	r	r	immediate data

Operation: MOV
(R_n) ← #data

MOV direct,A

Bytes: 2

Cycles: 1

Encoding:

1	1	1	1	0	1	0	1	direct address

Operation: MOV
(direct) ← (A)

MOV direct,R_n

Bytes: 2

Cycles: 2

Encoding:

1	0	0	0	1	r	r	r	direct address

Operation: MOV
(direct) ← (R_n)

MOV direct,direct

Bytes: 3

Cycles: 2

Encoding:

1	0	0	0	0	1	0	1	dir. addr. (scr)	dir. addr. (dest)

Operation: MOV
(direct) ← (direct)

MOV direct,@R_i

Bytes: 2

Cycles: 2

Encoding:

1	0	0	0	0	1	1	i	direct addr.

Operation: MOV
(direct) ← ((R_i))

MOV direct,#data

Bytes: 3

Cycles: 2

Encoding:

0	1	1	1	0	1	0	1	direct address	immediate data

Operation: MOV
(direct) ← #data

MOV @R_i,A

Bytes: 1

Cycles: 1

Encoding:

1	1	1	1	0	1	1	i

Operation: MOV
((R_i)) ← (A)

MOV @R_i,direct

Bytes: 2

Cycles: 2

Encoding:

1	0	1	0	0	1	1	i	direct addr.

Operation: MOV
((R_i)) ← (direct)

MOV @R_i,#data

Bytes: 2

Cycles: 1

Encoding:

0	1	1	1	0	1	1	i	immediate data

Operation: MOV
((R_i)) ← #data

MOV <dest-bit>,<src-bit>

Function: Move bit data

Description: MOV <dest-bit>,<src-bit> copies the Boolean variable indicated by the second operand into the location specified by the first operand. One of the operands must be the carry flag; the other may be any directly addressable bit. No other register or flag is affected.

Example: The carry flag is originally set. The data present at input Port 3 is 11000101B. The data previously written to output Port 1 is 35H (00110101B).

```
MOV     P1.3,C
MOV     C,P3.3
MOV     P1.2,C
```

leaves the carry cleared and changes Port 1 to 39H (00111001B).

MOV C,bit

Bytes: 2

Cycles: 1

Encoding:

1	0	1	0	0	0	1	0	bit address

Operation: MOV
(C) ← (bit)

MOV bit,C

Bytes: 2

Cycles: 2

Encoding:

1	0	0	1	0	0	1	0	bit address

Operation: MOV
(bit) ← (C)

MOV DPTR,#data16

Function: Load Data Pointer with a 16-bit constant

Description: MOV DPTR,#data16 loads the Data Pointer with the 16-bit constant indicated. The 16-bit constant is loaded into the second and third bytes of the instruction. The second byte (DPH) is the high-order byte, while the third byte (DPL) holds the lower-order byte. No flags are affected.

This is the only instruction which moves 16 bits of data at once.

Example: The instruction,

```
MOV     DPTR, # 1234H
```

loads the value 1234H into the Data Pointer: DPH holds 12H, and DPL holds 34H.

Bytes: 3

Cycles: 2

Encoding:

1	0	0	1	0	0	0	0	immed. data15-8	immed. data7-0

Operation: MOV
(DPTR) ← $\#data_{15-0}$
DPH ← DPL ← $\#data_{15-8}$ ← $\#data_{7-0}$

MOVC A,@A+ <base-reg>

Function: Move Code byte

Description: The MOVC instructions load the Accumulator with a code byte or constant from program memory. The address of the byte fetched is the sum of the original unsigned 8-bit Accumulator contents and the contents of a 16-bit base register, which may be either the Data Pointer or the PC. In the latter case, the PC is incremented to the address of the following instruction before being added with the Accumulator; otherwise the base register is not altered. Sixteen-bit addition is performed so a carry-out from the low-order eight bits may propagate through higher-order bits. No flags are affected.

Example: A value between 0 and 3 is in the Accumulator. The following instructions will translate the value in the Accumulator to one of four values defined by the DB (define byte) directive.

```
REL_PC:  INC   A
         MOVC  A,@A+PC
         RET
         DB    66H
         DB    77H
         DB    88H
         DB    99H
```

If the subroutine is called with the Accumulator equal to 01H, it returns with 77H in the Accumulator. The INC A before the MOVC instruction is needed to "get around" the RET instruction above the table. If several bytes of code separate the MOVC from the table, the corresponding number is added to the Accumulator instead.

MOVC A,@A+DPTR

Bytes: 1

Cycles: 2

Encoding:

1	0	0	1	0	0	1	1

Operation: MOVC
(A) ← ((A) + (DPTR))

MOVC A,@A+PC

Bytes: 1

Cycles: 2

Encoding:

1	0	0	0	0	0	1	1

Operation: MOVC
(PC) ← (PC) + 1
(A) ← ((A) + (PC))

MOVX <dest-byte>,<src-byte>

Function: Move External

Description: The MOVX instructions transfer data between the Accumulator and a byte of external data memory, which is why "X" is appended to MOV. There are two types of instructions, differing in whether they provide an 8-bit or 16-bit indirect address to the external data RAM.

In the first type, the contents of R0 or R1 in the current register bank provide an 8-bit address multiplexed with data on P0. Eight bits are sufficient for external I/O expansion decoding or for a relatively small RAM array. For somewhat larger arrays, any output port pins can be used to output higher-order address bits. These pins are controlled by an output instruction preceding the MOVX.

In the second type of MOVX instruction, the Data Pointer generates a 16-bit address. P2 outputs the high-order eight address bits (the contents of DPH), while P0 multiplexes the low-order eight bits (DPL) with data. The P2 Special Function Register retains its previous contents, while the P2 output buffers emit the contents of DPH. This form of MOVX is faster and more efficient when accessing very large data arrays (up to 64K bytes), since no additional instructions are needed to set up the output ports.

It is possible to use both MOVX types in some situations. A large RAM array with its high-order address lines driven by P2 can be addressed via the Data Pointer, or with code to output high-order address bits to P2, followed by a MOVX instruction using R0 or R1.

Example: An external 256 byte RAM using multiplexed address/data lines is connected to the 8051 Port 0. Port 3 provides control lines for the external RAM. Ports 1 and 2 are used for normal I/O. Registers 0 and 1 contain 12H and 34H. Location 34H of the external RAM holds the value 56H. The instruction sequence,

```
MOVX    A,@R1
MOVX    @R0,A
```

copies the value 56H into both the Accumulator and external RAM location 12H.

MOVX A,@R_i

Bytes: 1

Cycles: 2

Encoding:

1	1	1	0	0	0	1	i

Operation: MOVX
(A) ← ((Ri))

MOVX A,@DPTR

Bytes: 1

Cycles: 2

Encoding:

1	1	1	0	0	0	0	0

Operation: MOVX
(A) ← ((DPTR))

MOVX @R_i,A

Bytes: 1

Cycles: 2

Encoding:

1	1	1	1	0	0	1	i

Operation: MOVX
$((R_i)) \leftarrow (A)$

MOVX @DPTR,A

Bytes: 1

Cycles: 2

Encoding:

1	1	1	1	0	0	0	0

Operation: MOVX
(DPTR) ← (A)

MUL AB

Function: Multiply

Description: MUL AB multiplies the unsigned 8-bit integers in the Accumulator and register B. The low-order byte of the 16-bit product is left in the Accumulator, and the high-order byte in B. If the product is greater than 255 (0FFH), the overflow flag is set; otherwise it is cleared. The carry flag is always cleared.

Example: Originally the Accumulator holds the value 80 (50H). Register B holds the value 160 (0A0H). The instruction,

```
MUL     AB
```

will give the product 12,800 (3200H), so B is changed to 32H (00110010B) and the Accumulator is cleared. The overflow flag is set, carry is cleared.

Bytes: 1

Cycles: 4

Encoding:

1	0	1	0	0	1	0	0

Operation: MUL
$(A)_{7\text{-}0} \leftarrow (A) \times (B)$
$(B)_{15\text{-}8}$

NOP

Function: No Operation

Description: Execution continues at the following instruction. Other than the PC, no registers or flags are affected.

Example: A low-going output pulse on bit 7 of Port 2 must last exactly 5 cycles. A simple SETB/CLR sequence generates a one-cycle pulse, so four additional cycles must be inserted. This may be done (assuming no interrupts are enabled) with the following instruction sequence,

```
CLR     P2.7
NOP
NOP
NOP
NOP
SETB    P2.7
```

Bytes: 1

Cycles: 1

Encoding:

0	0	0	0	0	0	0	0

Operation: NOP
(PC) ← (PC) + 1

ORL <dest-byte> <src-byte>

Function: Logical-OR for byte variables

Description: ORL performs the bitwise logical-OR operation between the indicated variables, storing the results in the destination byte. No flags are affected.

The two operands allow six addressing mode combinations. When the destination is the Accumulator, the source can use register, direct, register-indirect, or immediate addressing; when the destination is a direct address, the source can be the Accumulator or immediate data.

Note: When this instruction is used to modify an output port, the value used as the original port data is read from the output data latch, *not* the input pins.

Example: If the Accumulator holds 0C3H (11000011B) and R0 holds 55H (01010101B) then the following instruction,

```
ORL     A,R0
```

leaves the Accumulator holding the value 0D7H (1101011IB).When the destination is a directly addressed byte, the instruction can set combinations of bits in any RAM location or hardware register. The pattern of bits to be set is determined by a mask byte, which may be either a constant data value in the instruction or a variable computed in the Accumulator at run-time. The instruction,

```
ORL     P1,#00110010B
```

sets bits 5, 4, and 1 of output Port 1.

ORL A,R_n

Bytes: 1

Cycles: 1

Encoding:

0	1	0	0	1	r	r	r

Operation: ORL
$(A) \leftarrow (A) \vee (R_n)$

ORL A,direct

Bytes: 2

Cycles: 1

Encoding:

0	1	0	0	0	1	0	1	direct address

Operation: ORL
(A) ← (A) ∨ (direct)

ORL A,@R_i

Bytes: 1

Cycles: 1

Encoding:

0	1	0	0	0	1	1	i

Operation: ORL
(A) ← (A) ∨((R$_i$))

ORL A,#data

Bytes: 2

Cycles: 1

Encoding:

0	1	0	0	0	1	0	0	immediate data

Operation: ORL
(A) ← (A) ∨ #data

ORL direct,A

Bytes: 2

Cycles: 1

Encoding:

0	1	0	0	0	0	1	0	direct address

Operation: ORL
(direct) ← (direct) ∨ (A)

ORL direct,#data

Bytes: 3

Cycles: 2

Encoding:

0	1	0	0	0	0	1	1	direct addr.	immediate data

Operation: ORL
(direct) ← (direct) ∨ #data

ORL C,<src-bit>

Function: Logical-OR for bit variables

Description: Set the carry flag if the Boolean value is a logical 1; leave the carry in its current state otherwise. A slash (/) preceding the operand in the assembly language indicates that the logical complement of the addressed bit is used as the source value, but the source bit itself is not affected. No other flags are affected.

Example: Set the carry flag if and only if P1.0 = 1, ACC. 7 = 1, or OV = 0:

```
MOV     C,P1.0      ;LOAD CARRY WITH INPUT PIN P10
ORL     C,ACC.7     ;OR CARRY WITH THE ACC. BIT 7
ORL     C,/OV       ;OR CARRY WITH THE INVERSE OF OV.
```

ORL C,bit

Bytes: 2

Cycles: 2

Encoding:

0	1	1	1	0	0	1	0	bit address

Operation: ORL
(C) ← (C) ∨ (bit)

ORL C,/bit

Bytes: 2

Cycles: 2

Encoding:

1	0	1	0	0	0	0	0	bit address

Operation: ORL
(C) ← (C) ∨ $\overline{(bit)}$

POP direct

Function: Pop from stack.

Description: The contents of the internal RAM location addressed by the Stack Pointer is read, and the Stack Pointer is decremented by one. The value read is then transferred to the directly addressed byte indicated. No flags are affected.

Example: The Stack Pointer originally contains the value 32H, and internal RAM locations 30H through 32H contain the values 20H, 23H, and 01H, respectively. The following instruction sequence,

```
POP     DPH
POP     DPL
```

leaves the Stack Pointer equal to the value 30H and sets the Data Pointer to 0123H. At this point, the following instruction,

```
POP     SP
```

leaves the Stack Pointer set to 20H. In this special case, the Stack Pointer was decremented to 2FH before being loaded with the value popped (20H).

Bytes: 2

Cycles: 2

Encoding:

1	1	0	1	0	0	0	0	direct address

Operation: POP
(direct) ← ((SP))
(SP) ← (SP) - 1

PUSH direct

Function: Push onto stack

Description: The Stack Pointer is incremented by one. The contents of the indicated variable is then copied into the internal RAM location addressed by the Stack Pointer. Otherwise no flags are affected.

Example: On entering an interrupt routine, the Stack Pointer contains 09H. The Data Pointer holds the value 0123H. The following instruction sequence,

```
PUSH    DPL
PUSH    DPH
```

leaves the Stack Pointer set to 0BH and stores 23H and 01H in internal RAM locations 0AH and 0BH, respectively.

Bytes: 2

Cycles: 2

Encoding:

1	1	0	0	0	0	0	0	direct address

Operation: PUSH
(SP) ← (SP) + 1
((SP)) ← (direct)

RET

Function: Return from subroutine

Description: RET pops the high- and low-order bytes of the PC successively from the stack, decrementing the Stack Pointer by two. Program execution continues at the resulting address, generally the instruction immediately following an ACALL or LCALL. No flags are affected.

Example: The Stack Pointer originally contains the value 0BH. Internal RAM locations 0AH and 0BH contain the values 23H and 01H, respectively. The following instruction,

```
RET
```

leaves the Stack Pointer equal to the value 09H. Program execution continues at location 0123H.

Bytes: 1

Cycles: 2

Encoding:

0	0	1	0	0	0	1	0

Operation: RET
$(PC_{15-8}) \leftarrow ((SP))$
$(SP) \leftarrow (SP) - 1$
$(PC_{7-0}) \leftarrow ((SP))$
$(SP) \leftarrow (SP) - 1$

RETI

Function: Return from interrupt

Description: RETI pops the high- and low-order bytes of the PC successively from the stack and restores the interrupt logic to accept additional interrupts at the same priority level as the one just processed. The Stack Pointer is left decremented by two. No other registers are affected; the PSW is *not* automatically restored to its pre-interrupt status. Program execution continues at the resulting address, which is generally the instruction immediately after the point at which the interrupt request was detected. If a lower- or same-level interrupt was pending when the RETI instruction is executed, that one instruction is executed before the pending interrupt is processed.

Example: The Stack Pointer originally contains the value 0BH. An interrupt was detected during the instruction ending at location 0122H. Internal RAM locations 0AH and 0BH contain the values 23H and 01H, respectively. The following instruction,

```
RETI
```

leaves the Stack Pointer equal to 09H and returns program execution to location 0123H.

Bytes: 1

Cycles: 2

Encoding:

0	0	1	1	0	0	1	0

Operation: RETI
$(PC_{15-8}) \leftarrow ((SP))$
$(SP) \leftarrow (SP) - 1$
$(PC_{7-0}) \leftarrow ((SP))$
$(SP) \leftarrow (SP) - 1$

RL A

Function: Rotate Accumulator Left

Description: The eight bits in the Accumulator are rotated one bit to the left. Bit 7 is rotated into the bit 0 position. No flags are affected.

Example: The Accumulator holds the value 0C5H (11000101B). The following instruction,

```
RL      A
```

leaves the Accumulator holding the value 8BH (10001011B) with the carry unaffected.

Bytes: 1

Cycles: 1

Encoding:

0	0	1	0	0	0	1	1

Operation: RL
$(A_{n+1}) \leftarrow (A_n)$ n = 0 - 6
$(A_0) \leftarrow (A_7)$

RLC A

Function: Rotate Accumulator Left through the Carry flag

Description: The eight bits in the Accumulator and the carry flag are together rotated one bit to the left. Bit 7 moves into the carry flag; the original state of the carry flag moves into the bit 0 position. No other flags are affected.

Example: The Accumulator holds the value 0C5H(11000101B), and the carry is zero. The following instruction,

```
RLC     A
```

leaves the Accumulator holding the value 8BH (10001010B) with the carry set.

Bytes: 1

Cycles: 1

Encoding:

0	0	1	1	0	0	1	1

Operation: RLC
$(A_{n+1}) \leftarrow (A_n)$ n = 0 - 6
$(A_0) \leftarrow (C)$
$(C) \leftarrow (A_7)$

RR A

Function: Rotate Accumulator Right

Description: The eight bits in the Accumulator are rotated one bit to the right. Bit 0 is rotated into the bit 7 position. No flags are affected.

Example: The Accumulator holds the value 0C5H (11000101B). The following instruction,

```
RR      A
```

leaves the Accumulator holding the value 0E2H (11100010B) with the carry unaffected.

Bytes: 1

Cycles: 1

Encoding:

0	0	0	0	0	0	1	1

Operation: RR
$(A_n) \leftarrow (A_{n+1})$ n = 0 - 6
$(A_7) \leftarrow (A_0)$

RRC A

Function: Rotate Accumulator Right through Carry flag

Description: The eight bits in the Accumulator and the carry flag are together rotated one bit to the right. Bit 0 moves into the carry flag; the original value of the carry flag moves into the bit 7 position. No other flags are affected.

Example: The Accumulator holds the value 0C5H (11000101B), the carry is zero. The following instruction,

```
RRC     A
```

leaves the Accumulator holding the value 62 (01100010B) with the carry set.

Bytes: 1

Cycles: 1

Encoding:

0	0	0	1	0	0	1	1

Operation: RRC
$(A_n) \leftarrow (A_{n+1})$ n = 0 - 6
$(A_7) \leftarrow (C)$
$(C) \leftarrow (A_0)$

SETB <bit>

Function: Set Bit

Description: SETB sets the indicated bit to one. SETB can operate on the carry flag or any directly addressable bit. No other flags are affected.

Example: The carry flag is cleared. Output Port 1 has been written with the value 34H (00110100B). The following instructions,

```
SETB    C
SETB    P1.0
```

sets the carry flag to 1 and changes the data output on Port 1 to 35H (00110101B).

SETB C

Bytes: 1

Cycles: 1

Encoding:

1	1	0	1	0	0	1	1

Operation: SETB
$(C) \leftarrow 1$

SETB bit

Bytes: 2

Cycles: 1

Encoding:

1	1	0	1	0	0	1	0	bit address

Operation: SETB
(bit) ← 1

SJMP rel

Function: Short Jump

Description: Program control branches unconditionally to the address indicated. The branch destination is computed by adding the signed displacement in the second instruction byte to the PC, after incrementing the PC twice. Therefore, the range of destinations allowed is from 128 bytes preceding this instruction 127 bytes following it.

Example: The label RELADR is assigned to an instruction at program memory location 0123H. The following instruction,

```
SJMP    RELADR
```

assembles into location 0100H. After the instruction is executed, the PC contains the value 0123H.

Note: Under the above conditions the instruction following SJMP is at 102H. Therefore, the displacement byte of the instruction is the relative offset (0123H-0102H) = 21H. Put another way, an SJMP with a displacement of 0FEH is a one-instruction infinite loop.

Bytes: 2

Cycles: 2

Encoding:

1	0	0	0	0	0	0	0	rel. address

Operation: SJMP
$(PC) \leftarrow (PC) + 2$
$(PC) \leftarrow (PC) + rel$

SUBB A,<src-byte>

Function: Subtract with borrow

Description: SUBB subtracts the indicated variable and the carry flag together from the Accumulator, leaving the result in the Accumulator. SUBB sets the carry (borrow) flag if a borrow is needed for bit 7 and clears C otherwise. (If C was set *before* executing a SUBB instruction, this indicates that a borrow was needed for the previous step in a multiple-precision subtraction, so the carry is subtracted from the Accumulator along with the source operand.) AC is set if a borrow is needed for bit 3 and cleared otherwise. OV is set if a borrow is needed into bit 6, but not into bit 7, or into bit 7, but not bit 6.

When subtracting signed integers, OV indicates a negative number produced when a negative value is subtracted from a positive value, or a positive result when a positive number is subtracted from a negative number.

The source operand allows four addressing modes: register, direct, register-indirect, or immediate.

Example: The Accumulator holds 0C9H (11001001B), register 2 holds 54H (01010100B), and the carry flag is set. The instruction,

SUBB A,R2

will leave the value 74H (01110100B) in the accumulator, with the carry flag and AC cleared but OV set.

Notice that 0C9H minus 54H is 75H. The difference between this and the above result is due to the carry (borrow) flag being set before the operation. If the state of the carry is not known before starting a single or multiple-precision subtraction, it should be explicitly cleared by CLR C instruction.

SUBB A,R_n

Bytes: 1

Cycles: 1

Encoding:

1	0	0	1	1	r	r	r

Operation: SUBB
(A) ← (A) - (C) - (R_n)

SUBB A,direct

Bytes: 2

Cycles: 1

Encoding:

1	0	0	1	0	1	0	1	direct address

Operation: SUBB
(A) ← (A) - (C) - (direct)

SUBB A,@R_i

Bytes: 1

Cycles: 1

Encoding:

1	0	0	1	0	1	1	i

Operation: SUBB
(A) ← (A) - (C) - ((R_i))

SUBB A,#data

Bytes: 2

Cycles: 1

Encoding:

1	0	0	1	0	1	0	0	immediate data

Operation: SUBB
(A) ← (A) - (C) - #data

SWAP A

Function: Swap nibbles within the Accumulator

Description: SWAP A interchanges the low- and high-order nibbles (four-bit fields) of the Accumulator (bits 3 through 0 and bits 7 through 4). The operation can also be thought of as a 4-bit rotate instruction. No flags are affected.

Example: The Accumulator holds the value 0C5H (11000101B). The instruction,

SWAP A

leaves the Accumulator holding the value 5CH (01011100B).

Bytes: 1

Cycles: 1

Encoding:

1	1	0	0	0	1	0	0

Operation: SWAP
(A_{3-0}) D (A_{7-4})

XCH A,<byte>

Function: Exchange Accumulator with byte variable

Description: XCH loads the Accumulator with the contents of the indicated variable, at the same time writing the original Accumulator contents to the indicated variable. The source/destination operand can use register, direct, or register-indirect addressing.

Example: R0 contains the address 20H. The Accumulator holds the value 3FH (0011111IB). Internal RAM location 20H holds the value 75H (01110101B). The following instruction,

XCH A,@R0

leaves RAM location 20H holding the values 3FH (00111111B) and 75H (01110101B) in the accumulator.

XCH A,R_n

Bytes: 1

Cycles: 1

Encoding:

1	1	0	0	1	r	r	r

Operation: XCH
(A) D ((R_n)

XCH A,direct

Bytes: 2

Cycles: 1

Encoding:

1	1	0	0	0	1	0	1	direct address

Operation: XCH
(A) D (direct)

XCH A,@R_i

Bytes: 1

Cycles: 1

Encoding:

1	1	0	0	0	1	1	i

Operation: XCH
(A) D ((R_i))

XCHD A,@R_i

Function: Exchange Digit

Description: XCHD exchanges the low-order nibble of the Accumulator (bits 3 through 0), generally representing a hexadecimal or BCD digit, with that of the internal RAM location indirectly addressed by the specified register. The high-order nibbles (bits 7-4) of each register are not affected. No flags are affected.

Example: R0 contains the address 20H. The Accumulator holds the value 36H (00110110B). Internal RAM location 20H holds the value 75H (01110101B). The following instruction,

```
XCHD    A,@R0
```

leaves RAM location 20H holding the value 76H (01110110B) and 35H (00110101B) in the Accumulator.

Bytes: 1

Cycles: 1

Encoding:

1	1	0	1	0	1	1	i

Operation: XCHD
(A_{3-0}) ⊕ $((R_{i3-0}))$

XRL <dest-byte>,<src-byte>

Function: Logical Exclusive-OR for byte variables

Description: XRL performs the bitwise logical Exclusive-OR operation between the indicated variables, storing the results in the destination. No flags are affected.

The two operands allow six addressing mode combinations. When the destination is the Accumulator, the source can use register, direct, register-indirect, or immediate addressing; when the destination is a direct address, the source can be the Accumulator or immediate data.

Note: When this instruction is used to modify an output port, the value used as the original port data is read from the output data latch, *not* the input pins.

Example: If the Accumulator holds 0C3H (1100001lB) and register 0 holds 0AAH (10101010B) then the instruction,

```
XRL     A,R0
```

leaves the Accumulator holding the value 69H (01101001B).

When the destination is a directly addressed byte, this instruction can complement combinations of bits in any RAM location or hardware register. The pattern of bits to be complemented is then determined by a mask byte, either a constant contained in the instruction or a variable computed in the Accumulator at run-time. The following instruction,

```
XRL     P1,#00110001B
```

complements bits 5, 4, and 0 of output Port 1.

XRL A,R_n

Bytes: 1

Cycles: 1

Encoding:

0	1	1	0	1	r	r	r

Operation: XRL
(A) ← (A) ⊕ (R_n)

XRL A,direct

Bytes: 2

Cycles: 1

Encoding:

0	1	1	0	0	1	0	1	direct address

Operation: XRL
(A) ← (A) ⊕ (direct)

XRL A,@R_i

Bytes: 1

Cycles: 1

Encoding:

0	1	1	0	0	1	1	i

Operation: XRL
(A) ← (A) ⊕ ((R_i))

XRL A,#data

Bytes: 2

Cycles: 1

Encoding:

0	1	1	0	0	1	0	0	immediate data

Operation: XRL
(A) ← (A) ⊕ #data

XRL direct,A

Bytes: 2

Cycles: 1

Encoding:

0	1	1	0	0	0	1	0	direct address

Operation: XRL
(direct) ← (direct) ⊕ (A)

XRL direct,#data

Bytes: 3

Cycles: 2

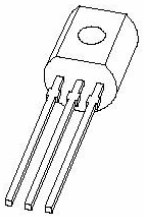
Encoding:

0	1	1	0	0	0	1	1	direct address	immediate data

Operation: XRL
(direct) ← (direct) ⊕ #data

DISCRETE SEMICONDUCTORS

DATA SHEET



BC546; BC547; BC548
NPN general purpose transistors

Product specification
Supersedes data of September 1994
File under Discrete Semiconductors, SC04

1997 Mar 04

PHILIPS

FEATURES

- Low current (max. 100 mA)
- Low voltage (max. 65 V).

APPLICATIONS

- General purpose switching and amplification.

DESCRIPTION

NPN transistor in a TO-92; SOT54 plastic package.
PNP complements: BC556, BC557 and BC558.

PINNING

PIN	DESCRIPTION
1	emitter
2	base
3	collector

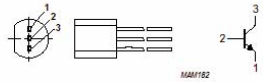


Fig.1 Simplified outline (TO-92; SOT54) and symbol.

QUICK REFERENCE DATA

SYMBOL	PARAMETER	CONDITIONS	MIN.	MAX.	UNIT
V_{CBO}	collector-base voltage	open emitter			
	BC546		–	80	V
	BC547		–	50	V
	BC548		–	30	V
V_{CEO}	collector-emitter voltage	open base			
	BC546		–	65	V
	BC547		–	45	V
	BC548		–	30	V
I_{CM}	peak collector current		–	200	mA
P_{tot}	total power dissipation	$T_{amb} \leq 25$ °C	–	500	mW
h_{FE}	DC current gain	I_C = 2 mA; V_{CE} = 5 V			
	BC546		110	450	
	BC547		110	800	
	BC548		110	800	
f_T	transition frequency	I_C = 10 mA; V_{CE} = 5 V; f = 100 MHz	100	–	MHz

LIMITING VALUES

In accordance with the Absolute Maximum Rating System (IEC 134).

SYMBOL	PARAMETER	CONDITIONS	MIN.	MAX.	UNIT
V_{CBO}	collector-base voltage	open emitter			
	BC546		–	80	V
	BC547		–	50	V
	BC548		–	30	V
V_{CEO}	collector-emitter voltage	open base			
	BC546		–	65	V
	BC547		–	45	V
	BC548		–	30	V
V_{EBO}	emitter-base voltage	open collector			
	BC546		–	6	V
	BC547		–	6	V
	BC548		–	5	V
I_C	collector current (DC)		–	100	mA
I_{CM}	peak collector current		–	200	mA
I_{BM}	peak base current		–	200	mA
P_{tot}	total power dissipation	$T_{amb} \leq 25$ °C; note 1	–	500	mW
T_{stg}	storage temperature		−65	+150	°C
T_j	junction temperature		–	150	°C
T_{amb}	operating ambient temperature		−65	+150	°C

Note

1. Transistor mounted on an FR4 printed-circuit board.

THERMAL CHARACTERISTICS

SYMBOL	PARAMETER	CONDITIONS	VALUE	UNIT
$R_{th\ j\text{-}a}$	thermal resistance from junction to ambient	note 1	0.25	K/mW

Note

1. Transistor mounted on an FR4 printed-circuit board.

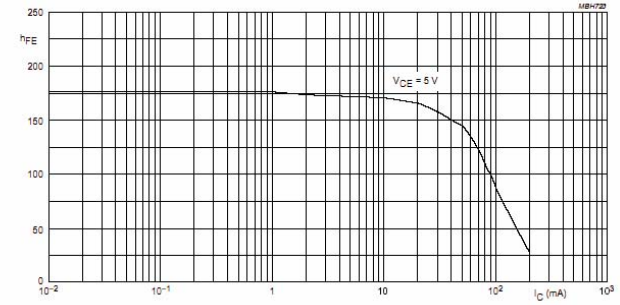
CHARACTERISTICS

$T_j = 25$ °C unless otherwise specified.

SYMBOL	PARAMETER	CONDITIONS	MIN.	TYP.	MAX.	UNIT
I_{CBO}	collector cut-off current	$I_E = 0$; $V_{CB} = 30$ V	–	–	15	nA
		$I_E = 0$; $V_{CB} = 30$ V; $T_j = 150$ °C	–	–	5	µA
I_{EBO}	emitter cut-off current	$I_C = 0$; $V_{EB} = 5$ V	–	–	100	nA
h_{FE}	DC current gain	$I_C = 10$ µA; $V_{CE} = 5$ V; see Figs 2, 3 and 4				
	BC546A; BC547A; BC548A		–	90	–	
	BC546B; BC547B; BC548B		–	150	–	
	BC547C; BC548C		–	270	–	
h_{FE}	DC current gain	$I_C = 2$ mA; $V_{CE} = 5$ V; see Figs 2, 3 and 4				
	BC546A; BC547A; BC548A		110	180	220	
	BC546B; BC547B; BC548B		200	290	450	
	BC547C; BC548C		420	520	800	
	BC547; BC548		110	–	800	
	BC546		110	–	450	
V_{CEsat}	collector-emitter saturation voltage	$I_C = 10$ mA; $I_B = 0.5$ mA	–	90	250	mV
		$I_C = 100$ mA; $I_B = 5$ mA	–	200	600	mV
V_{BEsat}	base-emitter saturation voltage	$I_C = 10$ mA; $I_B = 0.5$ mA; note 1	–	700	–	mV
		$I_C = 100$ mA; $I_B = 5$ mA; note 1	–	900	–	mV
V_{BE}	base-emitter voltage	$I_C = 2$ mA; $V_{CE} = 5$ V; note 2	580	660	700	mV
		$I_C = 10$ mA; $V_{CE} = 5$ V	–	–	770	mV
C_c	collector capacitance	$I_E = I_e = 0$; $V_{CB} = 10$ V; f = 1 MHz	–	1.5	–	pF
C_e	emitter capacitance	$I_C = I_c = 0$; $V_{EB} = 0.5$ V; f = 1 MHz	–	11	–	pF
f_T	transition frequency	$I_C = 10$mA; $V_{CE} = 5$ V; f = 100 MHz	100	–	–	MHz
F	noise figure	$I_C = 200$ µA; $V_{CE} = 5$ V; $R_S = 2$ kΩ; f = 1 kHz; B = 200 Hz	–	2	10	dB

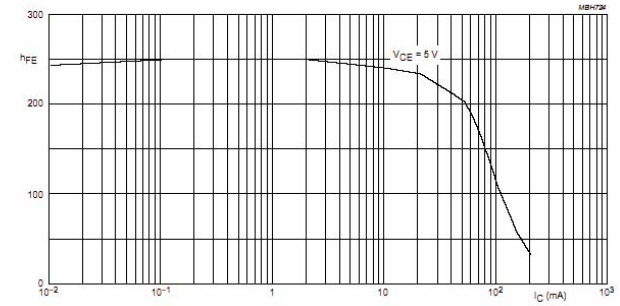
Notes

1. V_{BEsat} decreases by about 1.7 mV/K with increasing temperature.
2. V_{BE} decreases by about 2 mV/K with increasing temperature.



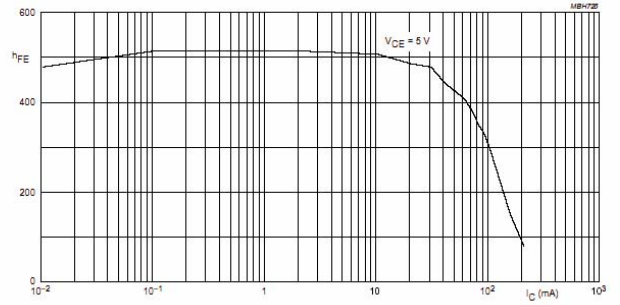
BC546A; BC547A; BC548A.

Fig.2 DC current gain; typical values.



BC546B; BC547B; BC548B.

Fig.3 DC current gain; typical values.

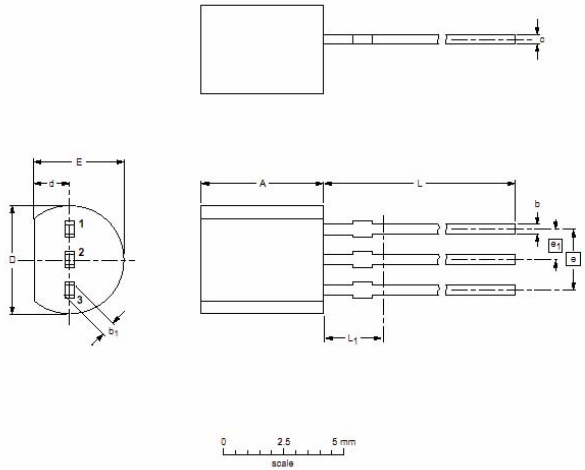


BC547C; BC548C.

Fig.4 DC current gain; typical values.

PACKAGE OUTLINE

Plastic single-ended leaded (through hole) package; 3 leads **SOT54**



DIMENSIONS (mm are the original dimensions)

UNIT	A	b	b_1	c	D	d	E	e	e_1	L	L_1(1)
mm	5.2 5.0	0.48 0.40	0.66 0.56	0.45 0.40	4.8 4.4	1.7 1.4	4.2 3.6	2.54	1.27	14.5 12.7	2.5

Note

1. Terminal dimensions within this zone are uncontrolled to allow for flow of plastic and terminal irregularities.

OUTLINE VERSION	REFERENCES				EUROPEAN PROJECTION	ISSUE DATE
	IEC	JEDEC	EIAJ			
SOT54		TO-92	SC-43			97-02-28

DEFINITIONS

Data Sheet Status	
Objective specification	This data sheet contains target or goal specifications for product development.
Preliminary specification	This data sheet contains preliminary data; supplementary data may be published later.
Product specification	This data sheet contains final product specifications.
Limiting values	
Limiting values given are in accordance with the Absolute Maximum Rating System (IEC 134). Stress above one or more of the limiting values may cause permanent damage to the device. These are stress ratings only and operation of the device at these or at any other conditions above those given in the Characteristics sections of the specification is not implied. Exposure to limiting values for extended periods may affect device reliability.	
Application information	
Where application information is given, it is advisory and does not form part of the specification.	

LIFE SUPPORT APPLICATIONS

These products are not designed for use in life support appliances, devices, or systems where malfunction of these products can reasonably be expected to result in personal injury. Philips customers using or selling these products for use in such applications do so at their own risk and agree to fully indemnify Philips for any damages resulting from such improper use or sale.

NOTES

Philips Semiconductors – a worldwide company

Argentina: see South America

Australia: 34 Waterloo Road, NORTH RYDE, NSW 2113,
Tel. +61 2 9805 4455, Fax. +61 2 9805 4466

Austria: Computerstr. 6, A-1101 WIEN, P.O. Box 213,
Tel. +43 1 60 101, Fax. +43 1 60 101 1210

Belarus: Hotel Minsk Business Center, Bld. 3, r. 1211, Volodarski Str. 6,
220050 MINSK, Tel. +375 172 200 733, Fax. +375 172 200 773

Belgium: see The Netherlands

Brazil: see South America

Bulgaria: Philips Bulgaria Ltd., Energoproject, 15th floor,
51 James Bourchier Blvd., 1407 SOFIA,
Tel. +359 2 689 211, Fax. +359 2 689 102

Canada: PHILIPS SEMICONDUCTORS/COMPONENTS,
Tel. +1 800 234 7381

China/Hong Kong: 501 Hong Kong Industrial Technology Centre,
72 Tat Chee Avenue, Kowloon Tong, HONG KONG,
Tel. +852 2319 7888, Fax. +852 2319 7700

Colombia: see South America

Czech Republic: see Austria

Denmark: Prags Boulevard 80, PB 1919, DK-2300 COPENHAGEN S,
Tel. +45 32 88 2636, Fax. +45 31 57 1949

Finland: Sinikalliontie 3, FIN-02630 ESPOO,
Tel. +358 9 615800, Fax. +358 9 61580/xxx

France: 4 Rue du Port-aux-Vins, BP317, 92156 SURESNES Cedex,
Tel. +33 1 40 99 6161, Fax. +33 1 40 99 6427

Germany: Hammerbrookstraße 69, D-20097 HAMBURG,
Tel. +49 40 23 53 60, Fax. +49 40 23 536 300

Greece: No. 15, 25th March Street, GR 17778 TAVROS/ATHENS,
Tel. +30 1 4894 339/239, Fax. +30 1 4814 240

Hungary: see Austria

India: Philips INDIA Ltd, Shivsagar Estate, A Block, Dr. Annie Besant Rd.
Worli, MUMBAI 400 018, Tel. +91 22 4938 541, Fax. +91 22 4938 722

Indonesia: see Singapore

Ireland: Newstead, Clonskeagh, DUBLIN 14,
Tel. +353 1 7640 000, Fax. +353 1 7640 200

Israel: RAPAC Electronics, 7 Kehilat Saloniki St, TEL AVIV 61180,
Tel. +972 3 645 0444, Fax. +972 3 649 1007

Italy: PHILIPS SEMICONDUCTORS, Piazza IV Novembre 3,
20124 MILANO, Tel. +39 2 6752 2531, Fax. +39 2 6752 2557

Japan: Philips Bldg 13-37, Kohnan 2-chome, Minato-ku, TOKYO 108,
Tel. +81 3 3740 5130, Fax. +81 3 3740 5077

Korea: Philips House, 260-199 Itaewon-dong, Yongsan-ku, SEOUL,
Tel. +82 2 709 1412, Fax. +82 2 709 1415

Malaysia: No. 76 Jalan Universiti, 46200 PETALING JAYA, SELANGOR,
Tel. +60 3 750 5214, Fax. +60 3 757 4880

Mexico: 5900 Gateway East, Suite 200, EL PASO, TEXAS 79905,
Tel. +9-5 800 234 7381

Middle East: see Italy

Netherlands: Postbus 90050, 5600 PB EINDHOVEN, Bldg. VB,
Tel. +31 40 27 82785, Fax. +31 40 27 88399

New Zealand: 2 Wagener Place, C.P.O. Box 1041, AUCKLAND,
Tel. +64 9 849 4160, Fax. +64 9 849 7811

Norway: Box 1, Manglerud 0612, OSLO,
Tel. +47 22 74 8000, Fax. +47 22 74 8341

Philippines: Philips Semiconductors Philippines Inc.,
106 Valero St. Salcedo Village, P.O. Box 2108 MCC, MAKATI,
Metro MANILA, Tel. +63 2 816 6380, Fax. +63 2 817 3474

Poland: Ul. Lukiska 10, PL 04-123 WARSZAWA,
Tel. +48 22 612 2831, Fax. +48 22 612 2327

Portugal: see Spain

Romania: see Italy

Russia: Philips Russia, Ul. Usatcheva 35A, 119048 MOSCOW,
Tel. +7 095 755 6918, Fax. +7 095 755 6919

Singapore: Lorong 1, Toa Payoh, SINGAPORE 1231,
Tel. +65 350 2538, Fax. +65 251 6500

Slovakia: see Austria

Slovenia: see Italy

South Africa: S.A. PHILIPS Pty Ltd., 195-215 Main Road Martindale,
2092 JOHANNESBURG, P.O. Box 7430 Johannesburg 2000,
Tel. +27 11 470 5911, Fax. +27 11 470 5494

South America: Rua do Rocio 220, 5th floor, Suite 51,
04552-903 São Paulo, SÃO PAULO - SP, Brazil,
Tel. +55 11 821 2333, Fax. +55 11 829 1849

Spain: Balmes 22, 08007 BARCELONA,
Tel. +34 3 301 6312, Fax. +34 3 301 4107

Sweden: Kottbygatan 7, Akalla, S-16485 STOCKHOLM,
Tel. +46 8 632 2000, Fax. +46 8 632 2745

Switzerland: Allmendstrasse 140, CH-8027 ZÜRICH,
Tel. +41 1 488 2686, Fax. +41 1 481 7730

Taiwan: Philips Semiconductors, 6F, No. 96, Chien Kuo N. Rd., Sec. 1,
TAIPEI, Taiwan Tel. +886 2 2134 2870, Fax. +886 2 2134 2874

Thailand: PHILIPS ELECTRONICS (THAILAND) Ltd.,
209/2 Sanpavuth-Bangna Road Prakanong, BANGKOK 10260,
Tel. +66 2 745 4090, Fax. +66 2 398 0793

Turkey: Talatpasa Cad. No. 5, 80640 GÜLTEPE/ISTANBUL,
Tel. +90 212 279 2770, Fax. +90 212 282 6707

Ukraine: PHILIPS UKRAINE, 4 Patrice Lumumba str., Building B, Floor 7,
252042 KIEV, Tel. +380 44 264 2776, Fax. +380 44 268 0461

United Kingdom: Philips Semiconductors Ltd., 276 Bath Road, Hayes,
MIDDLESEX UB3 5BX, Tel. +44 181 730 5000, Fax. +44 181 754 8421

United States: 811 East Arques Avenue, SUNNYVALE, CA 94088-3409,
Tel. +1 800 234 7381

Uruguay: see South America

Vietnam: see Singapore

Yugoslavia: PHILIPS, Trg N. Pasica 5/v, 11000 BEOGRAD,
Tel. +381 11 625 344, Fax.+381 11 635 777

For all other countries apply to: Philips Semiconductors, Marketing & Sales Communications,
Building BE-p, P.O. Box 218, 5600 MD EINDHOVEN, The Netherlands, Fax. +31 40 27 24825

Internet: http://www.semiconductors.philips.com

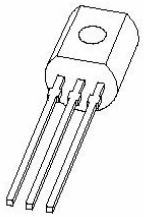
Printed in The Netherlands 117047/00/02/pp12 Date of release: 1997 Mar 04 Document order number: 9397 750 01853

Let's make things better.

Philips Semiconductors

PHILIPS

DATA SHEET



BC559; BC560
PNP general purpose transistors

Product specification 1997 Jun 03
Supersedes data of 1997 Mar 14
File under Discrete Semiconductors, SC04

PHILIPS

FEATURES

- Low current (max. 100 mA)
- Low voltage (max. 45 V).

APPLICATIONS

- General purpose switching and amplification.

DESCRIPTION

PNP transistor in a TO-92; SOT54 plastic package.
NPN complements: BC549 and BC550.

PINNING

PIN	DESCRIPTION
1	emitter
2	base
3	collector

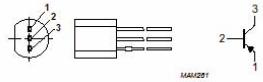


Fig.1 Simplified outline (TO-92; SOT54) and symbol.

QUICK REFERENCE DATA

SYMBOL	PARAMETER	CONDITIONS	MIN.	MAX.	UNIT
V_{CBO}	collector-base voltage	open emitter			
	BC559		–	–30	V
	BC560		–	–50	V
V_{CEO}	collector-emitter voltage	open base			
	BC559		–	–30	V
	BC560		–	–45	V
I_{CM}	peak collector current		–	–200	mA
P_{tot}	total power dissipation	$T_{amb} \leq 25$ °C	–	500	mW
h_{FE}	DC current gain	$I_C = -2$ mA; $V_{CE} = -5$ V	125	800	
f_T	transition frequency	$I_C = -10$ mA; $V_{CE} = -5$ V; f = 100 MHz	100	–	MHz

LIMITING VALUES

In accordance with the Absolute Maximum Rating System (IEC 134).

SYMBOL	PARAMETER	CONDITIONS	MIN.	MAX.	UNIT
V_{CBO}	collector-base voltage	open emitter			
	BC559		–	–30	V
	BC560		–	–50	V
V_{CEO}	collector-emitter voltage	open base			
	BC559		–	–30	V
	BC560		–	–45	V
V_{EBO}	emitter-base voltage	open collector	–	–5	V
I_C	collector current (DC)		–	–100	mA
I_{CM}	peak collector current		–	–200	mA
I_{BM}	peak base current		–	–200	mA
P_{tot}	total power dissipation	$T_{amb} \leq 25$ °C	–	500	mW
T_{stg}	storage temperature		–65	+150	°C
T_j	junction temperature		–	150	°C
T_{amb}	operating ambient temperature		–65	+150	°C

THERMAL CHARACTERISTICS

SYMBOL	PARAMETER	CONDITIONS	VALUE	UNIT
$R_{th\ j\text{-}a}$	thermal resistance from junction to ambient	note 1	250	K/W

Note

1. Transistor mounted on an FR4 printed-circuit board.

CHARACTERISTICS

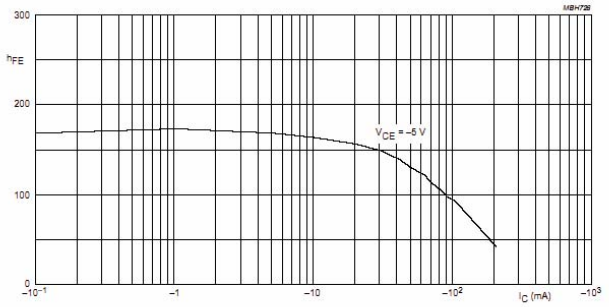
$T_j = 25$ °C unless otherwise specified.

SYMBOL	PARAMETER	CONDITIONS	MIN.	TYP.	MAX.	UNIT
I_{CBO}	collector cut-off current	$I_E = 0$; $V_{CB} = -30$ V	–	–1	–15	nA
		$I_E = 0$; $V_{CB} = -30$ V; $T_j = 150$ °C	–	–	–4	µA
I_{EBO}	emitter cut-off current	$I_C = 0$; $V_{EB} = -5$ V	–	–	–100	nA
h_{FE}	DC current gain	$I_C = -2$ mA; $V_{CE} = -5$ V; see Figs 2, 3 and 4	125	–	800	
h_{FE}	DC current gain	$I_C = -2$ mA; $V_{CE} = -5$ V; see Figs 2, 3 and 4				
	BC559A		125	–	250	
	BC559B; BC560B		220	–	475	
	BC559C; BC560C		420	–	800	
V_{CEsat}	collector-emitter saturation voltage	$I_C = -10$ mA; $I_B = -0.5$ mA	–	–60	–300	mV
		$I_C = -100$ mA; $I_B = -5$ mA	–	–180	–650	mV
V_{BEsat}	base-emitter saturation voltage	$I_C = -10$ mA; $I_B = -0.5$ mA; note 1	–	–750	–	mV
		$I_C = -100$ mA; $I_B = -5$ mA; note 1	–	–930	–	mV

SYMBOL	PARAMETER	CONDITIONS	MIN.	TYP.	MAX.	UNIT
V_{BE}	base-emitter voltage	$I_C = -2$ mA; $V_{CE} = -5$ V; note 2	–600	–650	–750	mV
		$I_C = -10$ mA; $V_{CE} = -5$ V; note 2	–	–	–820	mV
C_c	collector capacitance	$I_E = I_e = 0$; $V_{CB} = -10$ V; f = 1 MHz	–	4	–	pF
f_T	transition frequency	$I_E = -10$ mA; $V_{CB} = -5$ V; f = 100 MHz	100	–	–	MHz
F	noise figure	$I_C = -200$ µA; $V_{CE} = -5$ V; $R_S = 2$ kΩ; f = 30 Hz to 15.7 kHz				
	BC559A; BC560A		–	–	10	dB
	BC559B; BC560B; BC559C; BC560C		–	–	4	dB
F	noise figure	$I_C = -200$ µA; $V_{CE} = -5$ V; $R_S = 2$ kΩ; f = 1 kHz; B = 200 Hz				
	BC559A; BC560A		–	–	10	dB
	BC559B; BC560B; BC559C; BC560C		–	–	4	dB

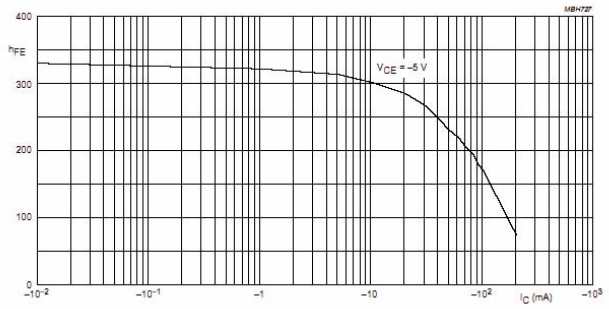
Notes

1. V_{BEsat} decreases by about –1.7 mV/K with increasing temperature.
2. V_{BE} decreases by about –2 mV/K with increasing temperature.



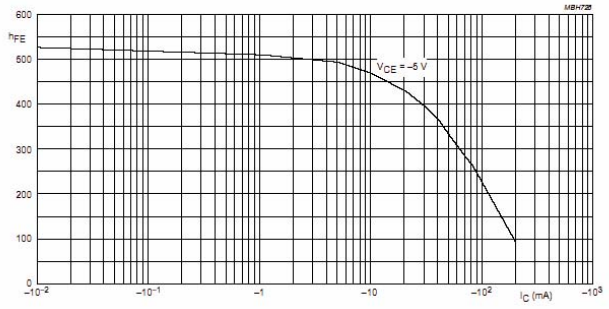
BC559A.

Fig.2 DC current gain; typical values.



BC559B; BC560B.

Fig.3 DC current gain; typical values.

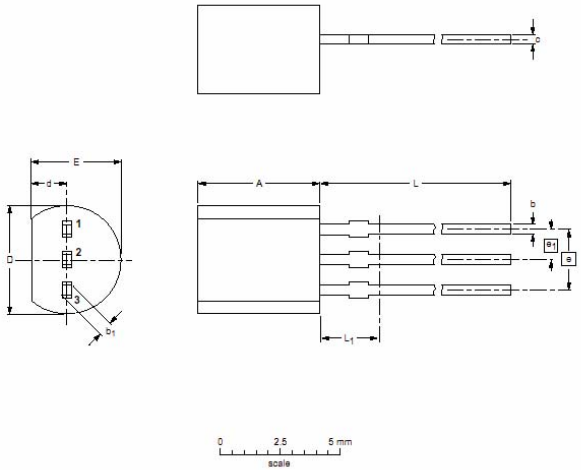


B559C; BC560C.

Fig.4 DC current gain; typical values.

PACKAGE OUTLINE

Plastic single-ended leaded (through hole) package; 3 leads **SOT54**



DIMENSIONS (mm are the original dimensions)

UNIT	A	b	b_1	c	D	d	E	e	e_1	L	L_1[1]
mm	5.2 5.0	0.48 0.40	0.66 0.56	0.45 0.40	4.8 4.4	1.7 1.4	4.2 3.6	2.54	1.27	14.5 12.7	2.5

Note

1. Terminal dimensions within this zone are uncontrolled to allow for flow of plastic and terminal irregularities.

OUTLINE VERSION	REFERENCES				EUROPEAN PROJECTION	ISSUE DATE
	IEC	JEDEC	EIAJ			
SOT54		TO-92	SC-43			97-02-28

DEFINITIONS

Data sheet status	
Objective specification	This data sheet contains target or goal specifications for product development.
Preliminary specification	This data sheet contains preliminary data; supplementary data may be published later.
Product specification	This data sheet contains final product specifications.
Limiting values	
Limiting values given are in accordance with the Absolute Maximum Rating System (IEC 134). Stress above one or more of the limiting values may cause permanent damage to the device. These are stress ratings only and operation of the device at these or at any other conditions above those given in the Characteristics sections of the specification is not implied. Exposure to limiting values for extended periods may affect device reliability.	
Application information	
Where application information is given, it is advisory and does not form part of the specification.	

LIFE SUPPORT APPLICATIONS

These products are not designed for use in life support appliances, devices, or systems where malfunction of these products can reasonably be expected to result in personal injury. Philips customers using or selling these products for use in such applications do so at their own risk and agree to fully indemnify Philips for any damages resulting from such improper use or sale.

Philips Semiconductors – a worldwide company

Argentina: see South America

Australia: 34 Waterloo Road, NORTH RYDE, NSW 2113,
Tel. +61 2 9805 4455, Fax. +61 2 9805 4466

Austria: Computerstr. 6, A-1101 WIEN, P.O. Box 213,
Tel. +43 1 60 101, Fax. +43 1 60 101 1210

Belarus: Hotel Minsk Business Center, Bld. 3, r. 1211, Volodarski Str. 6,
220050 MINSK, Tel. +375 172 200 733, Fax. +375 172 200 773

Belgium: see The Netherlands

Brazil: see South America

Bulgaria: Philips Bulgaria Ltd., Energoproject, 15th floor,
51 James Bourchier Blvd., 1407 SOFIA,
Tel. +359 2 689 211, Fax. +359 2 689 102

Canada: PHILIPS SEMICONDUCTORS/COMPONENTS,
Tel. +1 800 234 7381

China/Hong Kong: 501 Hong Kong Industrial Technology Centre,
72 Tat Chee Avenue, Kowloon Tong, HONG KONG,
Tel. +852 2319 7888, Fax. +852 2319 7700

Colombia: see South America

Czech Republic: see Austria

Denmark: Prags Boulevard 80, PB 1919, DK-2300 COPENHAGEN S,
Tel. +45 32 88 2636, Fax. +45 31 57 0044

Finland: Sinikalliontie 3, FIN-02630 ESPOO,
Tel. +358 9 615800, Fax. +358 9 61580920

France: 4 Rue du Port-aux-Vins, BP317, 92156 SURESNES Cedex,
Tel. +33 1 40 99 6161, Fax. +33 1 40 99 6427

Germany: Hammerbrookstraße 69, D-20097 HAMBURG,
Tel. +49 40 23 53 60, Fax. +49 40 23 536 300

Greece: No. 15, 25th March Street, GR 17778 TAVROS/ATHENS,
Tel. +30 1 4894 339/239, Fax. +30 1 4814 240

Hungary: see Austria

India: Philips INDIA Ltd, Shivsagar Estate, A Block, Dr. Annie Besant Rd.
Worli, MUMBAI 400 018, Tel. +91 22 4938 541, Fax. +91 22 4938 722

Indonesia: see Singapore

Ireland: Newstead, Clonskeagh, DUBLIN 14,
Tel. +353 1 7640 000, Fax. +353 1 7640 200

Israel: RAPAC Electronics, 7 Kehilat Saloniki St, PO Box 18053,
TEL AVIV 61180, Tel. +972 3 645 0444, Fax. +972 3 649 1007

Italy: PHILIPS SEMICONDUCTORS, Piazza IV Novembre 3,
20124 MILANO, Tel. +39 2 6752 2531, Fax. +39 2 6752 2557

Japan: Philips Bldg 13-37, Kohnan 2-chome, Minato-ku, TOKYO 108,
Tel. +81 3 3740 5130, Fax. +81 3 3740 5077

Korea: Philips House, 260-199 Itaewon-dong, Yongsan-ku, SEOUL,
Tel. +82 2 709 1412, Fax. +82 2 709 1415

Malaysia: No. 76 Jalan Universiti, 46200 PETALING JAYA, SELANGOR,
Tel. +60 3 750 5214, Fax. +60 3 757 4880

Mexico: 5900 Gateway East, Suite 200, EL PASO, TEXAS 79905,
Tel. +9-5 800 234 7381

Middle East: see Italy

Netherlands: Postbus 90050, 5600 PB EINDHOVEN, Bldg. VB,
Tel. +31 40 27 82785, Fax. +31 40 27 88399

New Zealand: 2 Wagener Place, C.P.O. Box 1041, AUCKLAND,
Tel. +64 9 849 4160, Fax. +64 9 849 7811

Norway: Box 1, Manglerud 0612, OSLO,
Tel. +47 22 74 8000, Fax. +47 22 74 8341

Philippines: Philips Semiconductors Philippines Inc.,
106 Valero St. Salcedo Village, P.O. Box 2108 MCC, MAKATI,
Metro MANILA, Tel. +63 2 816 6380, Fax. +63 2 817 3474

Poland: Ul. Lukiska 10, PL 04-123 WARSZAWA,
Tel. +48 22 612 2831, Fax. +48 22 612 2327

Portugal: see Spain

Romania: see Italy

Russia: Philips Russia, Ul. Usatcheva 35A, 119048 MOSCOW,
Tel. +7 095 755 6918, Fax. +7 095 755 6919

Singapore: Lorong 1, Toa Payoh, SINGAPORE 1231,
Tel. +65 350 2538, Fax. +65 251 6500

Slovakia: see Austria

Slovenia: see Italy

South Africa: S.A. PHILIPS Pty Ltd., 195-215 Main Road Martindale,
2092 JOHANNESBURG, P.O. Box 7430 Johannesburg 2000,
Tel. +27 11 470 5911, Fax. +27 11 470 5494

South America: Rua do Rocio 220, 5th floor, Suite 51,
04552-903 São Paulo, SÃO PAULO - SP, Brazil,
Tel. +55 11 821 2333, Fax. +55 11 829 1849

Spain: Balmes 22, 08007 BARCELONA,
Tel. +34 3 301 6312, Fax. +34 3 301 4107

Sweden: Kottbygatan 7, Akalla, S-16485 STOCKHOLM,
Tel. +46 8 632 2000, Fax. +46 8 632 2745

Switzerland: Allmendstrasse 140, CH-8027 ZÜRICH,
Tel. +41 1 488 2686, Fax. +41 1 481 7730

Taiwan: Philips Semiconductors, 6F, No. 96, Chien Kuo N. Rd., Sec. 1,
TAIPEI, Taiwan Tel. +886 2 2134 2865, Fax. +886 2 2134 2874

Thailand: PHILIPS ELECTRONICS (THAILAND) Ltd.,
209/2 Sanpavuth-Bangna Road Prakanong, BANGKOK 10260,
Tel. +66 2 745 4090, Fax. +66 2 398 0793

Turkey: Talatpasa Cad. No. 5, 80640 GÜLTEPE/ISTANBUL,
Tel. +90 212 279 2770, Fax. +90 212 282 6707

Ukraine: PHILIPS UKRAINE, 4 Patrice Lumumba str., Building B, Floor 7,
252042 KIEV, Tel. +380 44 264 2776, Fax. +380 44 268 0461

United Kingdom: Philips Semiconductors Ltd., 276 Bath Road, Hayes,
MIDDLESEX UB3 5BX, Tel. +44 181 730 5000, Fax. +44 181 754 8421

United States: 811 East Arques Avenue, SUNNYVALE, CA 94088-3409,
Tel. +1 800 234 7381

Uruguay: see South America

Vietnam: see Singapore

Yugoslavia: PHILIPS, Trg N. Pasica 5/v, 11000 BEOGRAD,
Tel. +381 11 625 344, Fax.+381 11 635 777

For all other countries apply to: Philips Semiconductors, Marketing & Sales Communications,
Building BE-p, P.O. Box 218, 5600 MD EINDHOVEN, The Netherlands, Fax. +31 40 27 24825

Internet: http://www.semiconductors.philips.com

Printed in The Netherlands 117047/00/03/pp8 Date of release: 1997 Jun 03 Document order number: 9397 750 01958

Let's make things better.

Philips Semiconductors

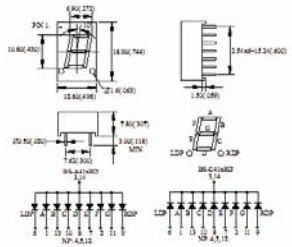
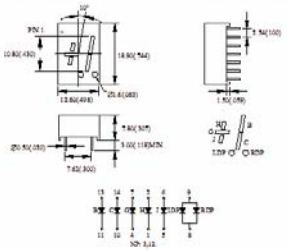
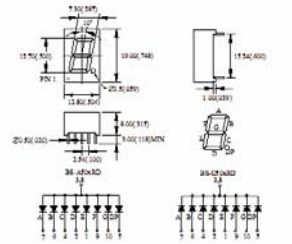
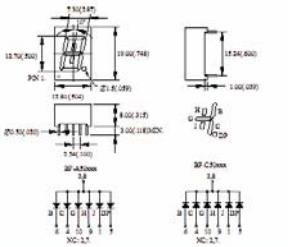
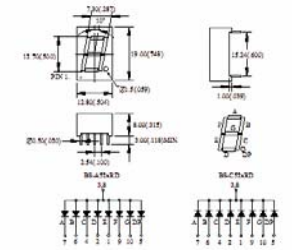
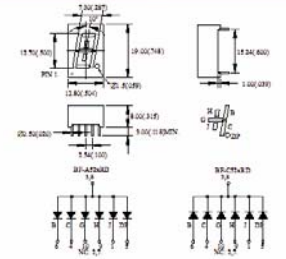
PHILIPS

SINGLE DIGIT LED DISPLAYS

Digit Size	Part No.		Chip		Absolute Maximum Ratings				Electro-optical Data(At 10mA)			Drawing No.
	Common Anode	Common Cathode	Material/Emitted Color	Peak Wave Length λp(nm)	Δλ (nm)	Pd (mw)	If (mA)	Ifp (mA)	Vf (v) Typ.	Vf (v) Max.	Iv. Typ. Per.Seg. (mcd)	
0.43" Single-Digit	BS-A411RD	BS-C411RD	GaAsP/Rad	655	40	80	40	200	1.7	2.0	0.7	SD-19
	BS-A415RD	BS-C415RD	GaP/ Bright Red	700	90	40	15	50	2.2	2.5	1.5	
	BS-A415RE	BS-C415RE										
	BS-A412RD	BS-C412RD	GaP/ Green	568	30	80	30	150	2.2	2.5	3.5	
	BS-A413RD	BS-C413RD	GaAsP/GaP/ Yellow	585	35	80	30	150	2.1	2.5	2.5	
	BS-A414RD	BS-C414RD	GaAsP/GaP/ Hi-Eff.Red GaAsP/GaP/ Orange	635	45	80	30	150	2.0	2.5	3.5	
	BS-A416RD	BS-C416RD	GaAlAs/ SH Super Red	660	20	80	30	150	1.7	2.5	6.5	
	BS-A41DRD	BS-C41DRD	GaAlAs/ DH Super Red	660	20	80	30	150	1.7	2.5	7.5	
0.43" Overflow Single-Digit	BF-U411RD		GaAsP/Rad	655	40	80	40	200	1.7	2.0	0.7	SD-20
	BF-U415RD		GaP/ Bright Red	700	90	40	15	50	2.2	2.5	1.5	
	BF-U415RE											
	BF-U412RD		GaP/ Green	568	30	80	30	150	2.2	2.5	3.5	
	BF-U413RD		GaAsP/GaP/ Yellow	585	35	80	30	150	2.1	2.5	2.5	
	BF-U414RD		GaAsP/GaP/ Hi-Eff.Red GaAsP/GaP/ Orange	635	45	80	30	150	2.0	2.5	3.5	
	BF-U416RD		GaAlAs/ SH Super Red	660	20	80	30	150	1.7	2.5	6.5	
	BF-U41DRD		GaAlAs/ DH Super Red	660	20	80	30	150	1.7	2.5	7.5	
0.50" Single-Digit	BS-A501RD	BS-C501RD	GaAsP/Rad	655	40	80	40	200	1.7	2.0	1.0	SD-21
	BS-A505RD	BS-C505RD	GaP/ Bright Red	700	90	40	15	50	2.2	2.5	1.5	
	BS-A505RE	BS-C505RE										
	BS-A502RD	BS-C502RD	GaP/ Green	568	30	80	30	150	2.2	2.5	3.0	
	BS-A503RD	BS-C503RD	GaAsP/GaP/ Yellow	585	35	80	30	150	2.1	2.5	2.0	
	BS-A504RD	BS-C504RD	GaAsP/GaP/ Hi-Eff.Red GaAsP/GaP/ Orange	635	45	80	30	150	2.0	2.5	3.0	
	BS-A506RD	BS-C506RD	GaAlAs/ SH Super Red	660	20	80	30	150	1.7	2.5	7.0	
	BS-A50DRD	BS-C50DRD	GaAlAs/ DH Super Red	660	20	80	30	150	1.7	2.5	8.0	
0.50" Overflow Single-Digit	BF-A501RD	BF-C501RD	GaAsP/Rad	655	40	80	40	200	1.7	2.0	1.0	SD-22
	BF-A505RD	BF-C505RD	GaP/ Bright Red	700	90	40	15	50	2.2	2.5	1.5	
	BF-A505RE	BF-C505RE										
	BF-A502RD	BF-C502RD	GaP/ Green	568	30	80	30	150	2.2	2.5	3.0	
	BF-A503RD	BF-C503RD	GaAsP/GaP/ Yellow	585	35	80	30	150	2.1	2.5	2.0	
	BF-A504RD	BF-C504RD	GaAsP/GaP/ Hi-Eff.Red GaAsP/GaP/ Orange	635	45	80	30	150	2.0	2.5	3.0	
	BF-A506RD	BF-C506RD	GaAlAs/ SH Super Red	660	20	80	30	150	1.7	2.5	7.0	
	BF-A50DRD	BF-C50DRD	GaAlAs/ DH Super Red	660	20	80	30	150	1.7	2.5	8.0	
0.50" Single-Digit	BS-A521RD	BS-C521RD	GaAsP/Rad	655	40	80	40	200	1.7	2.0	0.7	SD-23
	BS-A525RD	BS-C525RD	GaP/ Bright Red	700	90	40	15	50	2.2	2.5	1.5	
	BS-A525RE	BS-C525RE										
	BS-A522RD	BS-C522RD	GaP/ Green	568	30	80	30	150	2.2	2.5	3.5	
	BS-A523RD	BS-C523RD	GaAsP/GaP/ Yellow	585	35	80	30	150	2.1	2.5	2.5	
	BS-A524RD	BS-C524RD	GaAsP/GaP/ Hi-Eff.Red GaAsP/GaP/ Orange	635	45	80	30	150	2.0	2.5	3.5	
	BS-A526RD	BS-C526RD	GaAlAs/ SH Super Red	660	20	80	30	150	1.7	2.5	6.5	
	BS-A52DRD	BS-C52DRD	GaAlAs/ DH Super Red	660	20	80	30	150	1.7	2.5	7.5	
0.50" Overflow Single-Digit	BF-A521RD	BF-C521RD	GaAsP/Rad	655	40	80	40	200	1.7	2.0	0.7	SD-24
	BF-A525RD	BF-C525RD	GaP/ Bright Red	700	90	40	15	50	2.2	2.5	1.5	
	BF-A525RE	BF-C525RE										
	BF-A522RD	BF-C522RD	GaP/ Green	568	30	80	30	150	2.2	2.5	3.5	
	BF-A523RD	BF-C523RD	GaAsP/GaP/ Yellow	585	35	80	30	150	2.1	2.5	2.5	
	BF-A524RD	BF-C524RD	GaAsP/GaP/ Hi-Eff.Red GaAsP/GaP/ Orange	635	45	80	30	150	2.0	2.5	3.5	
	BF-A526RD	BF-C526RD	GaAlAs/ SH Super Red	660	20	80	30	150	1.7	2.5	6.5	
	BF-A52DRD	BF-C52DRD	GaAlAs/ DH Super Red	660	20	80	30	150	1.7	2.5	7.5	

Yellow Stone corp.
www.ystone.com.tw
886-2-28221522 FAX:886-2-28202309

SINGLE DIGIT LED DISPLAYS

SD-19	BS-$\frac{A}{C}$41xRD	SD-20	BF-U41xRD
			
SD-21	BS-$\frac{A}{C}$50xRD	SD-22	BF-$\frac{A}{C}$50xRD
			
SD-23	BS-$\frac{A}{C}$52xRD	SD-24	BF-$\frac{A}{C}$52xRD
			

NOTES: 1.All Dimensions are in millimeters(inches). 2.Tolerance is± 0.25mm(.010").
3.Specifications are subject to change without notice. 4.NP:No Pin. 5.NC:No Connect.

LAMPIRAN C

PERANGKAT LUNAK

```
0001   0000          ;-----------------------------------------------
0002   0000          ; PROGRAM PENGHITUNG TARIF KENDARAAN UMUM
0003   0000          ;-----------------------------------------------
0004   0000          #INCLUDE "8051.H"
0001+  0000           ;[]------------------------------------------------------------[]
0002+  0000           ;|                                                            |
0003+  0000           ;|                                                            |
0004+  0000           ;|          TASM 8051/8052 Equates header file                |
0005+  0000           ;|                                                            |
0006+  0000           ;|                                                            |
0007+  0000           ;|                                      August  1995          |
0008+  0000           ;[]------------------------------------------------------------[]
0009+  0000           #define ORG     .ORG
0010+  0000           #define END     .end
0011+  0000           #define equ     .equ
0012+  0000           #define data    .equ
0013+  0000           #define bit     .equ
0014+  0000
0015+  0000           P0      .equ    080H    ;Port 0 - Not present on the 89C2051
0016+  0000           SP      .equ    081H    ;Stack pointer
0017+  0000           DPL     .equ    082H    ;Data pointer low, part of 16 bit reg with DPH
0018+  0000           DPH     .equ    083H
0019+  0000           PCON    .equ    087H    ;Power control, not bit addressable,
0020+  0000           TCON    .equ    088H    ;Timer/counter control register, see bit list below
0021+  0000           TMOD    .equ    089H    ;Timer/counter mode control register
0022+  0000           TL0     .equ    08AH    ;Timer 0 low
0023+  0000           TL1     .equ    08BH    ;Timer 1 low
0024+  0000           TH0     .equ    08CH    ;Timer 0 high - also reload val in 8bit auto RL mode
0025+  0000           TH1     .equ    08DH    ;Timer 1 high - also reload val in 8bit auto RL mode
0026+  0000           P1      .equ    090H    ;Port 1
0027+  0000           SCON    .equ    098H    ;Serial port control register, see bit list below
0028+  0000           SBUF    .equ    099H    ;Serial buffer - read for Serial Rx, written to Tx
0029+  0000           P2      .equ    0A0H    ;Port 2 - Not present on 89C2051
0030+  0000           IE      .equ    0A8H    ;Interrupt enable register, see bit list below
0031+  0000           P3      .equ    0B0H    ;Port 3
0032+  0000           IP      .equ    0B8H    ;Interrupt priority register, see bit list below
0033+  0000           T2CON   .equ    0C8H    ;8052, 80154 only
0034+  0000           RCAP2L  .equ    0CAH    ;8052, 80154 only
0035+  0000           RCAP2H  .equ    0CBH    ;8052, 80154 only
0036+  0000           TL2     .equ    0CCH    ;8052, 80154 only
0037+  0000           TH2     .equ    0CDH    ;8052, 80154 only
0038+  0000           PSW     .equ    0D0H    ;Program status word, see bit list below
0039+  0000           ACC     .equ    0E0H    ;Accumulator
0040+  0000           B       .equ    0F0H    ;Secondary Accumulator, used in Multiply and Divide
0041+  0000           IOCON   .equ    0F8H    ;80154 only
0042+  0000
0043+  0000           ;PORT 0 BITS
0044+  0000           P0.0    .equ    080H    ;Port 0 bit 0
0045+  0000           P0.1    .equ    081H    ;Port 0 bit 1
0046+  0000           P0.2    .equ    082H    ;Port 0 bit 2
0047+  0000           P0.3    .equ    083H    ;Port 0 bit 3
0048+  0000           P0.4    .equ    084H    ;Port 0 bit 4
0049+  0000           P0.5    .equ    085H    ;Port 0 bit 5
0050+  0000           P0.6    .equ    086H    ;Port 0 bit 6
0051+  0000           P0.7    .equ    087H    ;Port 0 bit 7
0052+  0000
0053+  0000           ;PORT 1 BITS
0054+  0000           P1.0    .equ    090H    ;Port 1 bit 0
0055+  0000           P1.1    .equ    091H    ;Port 1 bit 1
0056+  0000           P1.2    .equ    092H    ;Port 1 bit 2
0057+  0000           P1.3    .equ    093H    ;Port 1 bit 3
0058+  0000           P1.4    .equ    094H    ;Port 1 bit 4
0059+  0000           P1.5    .equ    095H    ;Port 1 bit 5
0060+  0000           P1.6    .equ    096H    ;Port 1 bit 6
0061+  0000           P1.7    .equ    097H    ;Port 1 bit 7
0062+  0000
0063+  0000           ;PORT 2 BITS
0064+  0000           P2.0    .equ    0A0H    ;Port 2 bit 0
0065+  0000           P2.1    .equ    0A1H    ;Port 2 bit 1
```

```
0066+ 0000          P2.2    .equ    0A2H    ;Port 2 bit 2
0067+ 0000          P2.3    .equ    0A3H    ;Port 2 bit 3
0068+ 0000          P2.4    .equ    0A4H    ;Port 2 bit 4
0069+ 0000          P2.5    .equ    0A5H    ;Port 2 bit 5
0070+ 0000          P2.6    .equ    0A6H    ;Port 2 bit 6
0071+ 0000          P2.7    .equ    0A7H    ;Port 2 bit 7
0072+ 0000
0073+ 0000          ;PORT 3 BITS
0074+ 0000          P3.0    .equ    0B0H    ;Port 3 bit 0
0075+ 0000          P3.1    .equ    0B1H    ;Port 3 bit 1
0076+ 0000          P3.2    .equ    0B2H    ;Port 3 bit 2
0077+ 0000          P3.3    .equ    0B3H    ;Port 3 bit 3
0078+ 0000          P3.4    .equ    0B4H    ;Port 3 bit 4
0079+ 0000          P3.5    .equ    0B5H    ;Port 3 bit 5
0080+ 0000          P3.6    .equ    0B6H    ;Port 3 bit 6
0081+ 0000          P3.7    .equ    0B7H    ;Port 3 bit 7
0082+ 0000
0083+ 0000          ;ACCUMULATOR BITS
0084+ 0000          ACC.0   .equ    0E0H    ;Acc bit 0
0085+ 0000          ACC.1   .equ    0E1H    ;Acc bit 1
0086+ 0000          ACC.2   .equ    0E2H    ;Acc bit 2
0087+ 0000          ACC.3   .equ    0E3H    ;Acc bit 3
0088+ 0000          ACC.4   .equ    0E4H    ;Acc bit 4
0089+ 0000          ACC.5   .equ    0E5H    ;Acc bit 5
0090+ 0000          ACC.6   .equ    0E6H    ;Acc bit 6
0091+ 0000          ACC.7   .equ    0E7H    ;Acc bit 7
0092+ 0000
0093+ 0000          ;B REGISTER BITS
0094+ 0000          B.0     .equ    0F0H    ;Breg bit 0
0095+ 0000          B.1     .equ    0F1H    ;Breg bit 1
0096+ 0000          B.2     .equ    0F2H    ;Breg bit 2
0097+ 0000          B.3     .equ    0F3H    ;Breg bit 3
0098+ 0000          B.4     .equ    0F4H    ;Breg bit 4
0099+ 0000          B.5     .equ    0F5H    ;Breg bit 5
0100+ 0000          B.6     .equ    0F6H    ;Breg bit 6
0101+ 0000          B.7     .equ    0F7H    ;Breg bit 7
0102+ 0000
0103+ 0000          ;PSW REGISTER BITS
0104+ 0000          P       .equ    0D0H    ;Parity flag
0105+ 0000          F1      .equ    0D1H    ;User flag 1
0106+ 0000          OV      .equ    0D2H    ;Overflow flag
0107+ 0000          RS0     .equ    0D3H    ;Register bank select 1
0108+ 0000          RS1     .equ    0D4H    ;Register bank select 0
0109+ 0000          F0      .equ    0D5H    ;User flag 0
0110+ 0000          AC      .equ    0D6H    ;Auxiliary carry flag
0111+ 0000          CY      .equ    0D7H    ;Carry flag
0112+ 0000
0113+ 0000          ;TCON REGISTER BITS
0114+ 0000          IT0     .equ    088H    ;Intr 0 type control
0115+ 0000          IE0     .equ    089H    ;Intr 0 edge flag
0116+ 0000          IT1     .equ    08AH    ;Intr 1 type control
0117+ 0000          IE1     .equ    08BH    ;Intr 1 edge flag
0118+ 0000          TR0     .equ    08CH    ;Timer 0 run
0119+ 0000          TF0     .equ    08DH    ;Timer 0 overflow
0120+ 0000          TR1     .equ    08EH    ;Timer 1 run
0121+ 0000          TF1     .equ    08FH    ;Timer 1 overflow
0122+ 0000
0123+ 0000          ;SCON REGISTER BITS
0124+ 0000          RI      .equ    098H    ;RX Intr flag
0125+ 0000          TI      .equ    099H    ;TX Intr flag
0126+ 0000          RB8     .equ    09AH    ;RX 9th bit
0127+ 0000          TB8     .equ    09BH    ;TX 9th bit
0128+ 0000          REN     .equ    09CH    ;Enable RX flag
0129+ 0000          SM2     .equ    09DH    ;8/9 bit select flag
0130+ 0000          SM1     .equ    09EH    ;Serial mode bit 1
0131+ 0000          SM0     .equ    09FH    ;Serial mode bit 0
0132+ 0000
0133+ 0000          ;IE REGISTER BITS
0134+ 0000          EX0     .equ    0A8H    ;External intr 0
```

```
0135+ 0000          ET0     .equ    0A9H    ;Timer 0 intr
0136+ 0000          EX1     .equ    0AAH    ;External intr 1
0137+ 0000          ET1     .equ    0ABH    ;Timer 1 intr
0138+ 0000          ES      .equ    0ACH    ;Serial port intr
0139+ 0000          ET2     .equ    0ADH    ;Timer 2 intr
0140+ 0000          ;Reserved       0AEH    Reserved
0141+ 0000          EA      .equ    0AFH    ;Global intr enable
0142+ 0000
0143+ 0000          ;IP REGISTER BITS
0144+ 0000          PX0     .equ    0B8H    ;Priority level-External intr 0
0145+ 0000          PT0     .equ    0B9H    ;Priority level-Timer 0 intr
0146+ 0000          PX1     .equ    0BAH    ;Priority level-External intr 1
0147+ 0000          PT1     .equ    0BBH    ;Priority level-Timer 1 intr
0148+ 0000          PS      .equ    0BCH    ;Priority level-Serial port intr
0149+ 0000          PT2     .equ    0BDH    ;Priority level-Timer 2 intr
0150+ 0000          ;Reserved       0BEH    Reserved
0151+ 0000          PCT     .equ    0BFH    ;Global priority level
0152+ 0000
0153+ 0000          ;IOCON REGISTER BITS  80154 ONLY
0154+ 0000          ALF     .equ    0F8H    ;Power down port condition
0155+ 0000          P1HZ    .equ    0F9H    ;Port 1 control
0156+ 0000          P2HZ    .equ    0FAH    ;Port 2 control
0157+ 0000          P3HZ    .equ    0FBH    ;Port 3 control
0158+ 0000          IZC     .equ    0FCH    ;Pullup select
0159+ 0000          SERR    .equ    0FDH    ;Serial reception error
0160+ 0000          T32     .equ    0FEH    ;32 bit timer config
0161+ 0000          WDT     .equ    0FFH    ;Watchdog config
0162+ 0000
0163+ 0000          ;T2CON REGISTER BITS  8052/80154 ONLY
0164+ 0000          CP/RL2  .equ    0C8H    ;Timer 2 capture/reload flag
0165+ 0000          C/T2    .equ    0C9H    ;Timer 2 timer/counter select
0166+ 0000          TR2     .equ    0CAH    ;Timer 2 start/stop
0167+ 0000          EXEN2   .equ    0CBH    ;Timer 2 external enable
0168+ 0000          TCLK    .equ    0CCH    ;TX clock flag
0169+ 0000          RCLK    .equ    0CDH    ;RX clock flag
0170+ 0000          EXF2    .equ    0CEH    ;Timer 2 external flag
0171+ 0000          TF2     .equ    0CFH    ;Timer 2 overflow
0172+ 0000
0005  0000
0006  0000        DATAKARTU       .EQU    P1      ;P1.0 .. P1..3 KARTU NAIK
0007  0000                                ;P1.4 .. P1..6 KARTU TURUN
0008  0000        SENSORPIRING    .EQU    P3.2
0009  0000        DATA7SEG        .EQU    P0
0010  0000        SLCT7SEG1       .EQU    P2.0
0011  0000        SLCT7SEG2       .EQU    P2.1
0012  0000        SLCT7SEG3       .EQU    P2.2
0013  0000        SLCT7SEG4       .EQU    P2.3
0014  0000        SATUMETER       .EQU    2               ;1 METER= 2 PUTARAN PIRING
0015  0000        BSATUMETER1     .EQU    00H
0016  0000        BSATUMETER2     .EQU    08H             ;1 METER = RP. 8
0017  0000
0018  0030                .ORG    30H
0019  0030        B_PENUMPANG1_1  .BLOCK   1
0020  0031        B_PENUMPANG1_2  .BLOCK   1
0021  0032        B_PENUMPANG2_1  .BLOCK   1
0022  0033        B_PENUMPANG2_2  .BLOCK   1
0023  0034        B_PENUMPANG3_1  .BLOCK   1
0024  0035        B_PENUMPANG3_2  .BLOCK   1
0025  0036        B_PENUMPANG4_1  .BLOCK   1
0026  0037        B_PENUMPANG4_2  .BLOCK   1
0027  0038
0028  0038        BUFSEG1         .BLOCK   1
0029  0039        BUFSEG2         .BLOCK   1
0030  003A        BUFSEG3         .BLOCK   1
0031  003B        BUFSEG4         .BLOCK   1
0032  003C        JARAK1          .BLOCK   1
0033  003D        JARAK2          .BLOCK   1
0034  003E        BIAYA1          .BLOCK   1
0035  003F        BIAYA2          .BLOCK   1
```

```
0036  0040          HITUNGPUTARAN   .BLOCK   1          ;PENGHITUNG PUTARAN
0037  0041          METER           .BLOCK   1
0038  0042          FLAGPENUMPANG1  .BLOCK   1          ;TANDA PENUMPANG 1 - 4
0039  0043
0040  0043
0041  0043
0042  0000                      .ORG    00H
0043  0000 02 01 00               LJMP    START
0044  0003
0045  0100                      .ORG    100H
0046  0100 75 81 20    START:       MOV     SP,#20H
0047  0103 75 D0 00                MOV     PSW,#0
0048  0106 12 03 3D                LCALL   PROC_HAPUSSEMUABUFFER
0049  0109 75 40 00                MOV     HITUNGPUTARAN,#0
0050  010C 75 42 00                MOV     FLAGPENUMPANG1,#0
0051  010F 75 80 FF                MOV     DATA7SEG,#0FFH
0052  0112 75 41 00                MOV     METER,#0
0053  0115
0054  0115 75 81 20    LOOPING:     MOV     SP,#20H
0055  0118 12 01 4D                LCALL   PROC_NAIK
0056  011B 12 01 E8                LCALL   PROC_TURUN
0057  011E          CEKPASS:
0058  011E 75 38 BF                MOV     BUFSEG1,#0BFH
0059  0121 75 39 BF                MOV     BUFSEG2,#0BFH
0060  0124 75 3A BF                MOV     BUFSEG3,#0BFH
0061  0127 75 3B BF                MOV     BUFSEG4,#0BFH
0062  012A 12 02 E6                LCALL   SCANNING
0063  012D 30 B2 E5                JNB     SENSORPIRING,LOOPING
0064  0130 20 B2 FD    LLP:         JB      SENSORPIRING,LLP
0065  0133 E5 40                   MOV     A,HITUNGPUTARAN
0066  0135 24 01                   ADD     A,#01
0067  0137 D4                      DA      A
0068  0138 F5 40                   MOV     HITUNGPUTARAN,A
0069  013A B4 02 D8                CJNE    A,#SATUMETER,LOOPING
0070  013D 75 40 00                MOV     HITUNGPUTARAN,#0
0071  0140 E5 41                   MOV     A,METER
0072  0142 24 01                   ADD     A,#01
0073  0144 D4                      DA      A
0074  0145 F5 41                   MOV     METER,A
0075  0147 12 02 79                LCALL   PROC_HITUNG
0076  014A 02 01 15                LJMP    LOOPING
0077  014D
0078  014D          ;-------------------------------------------------
0079  014D          ; PROCEDURE NAIK
0080  014D          ;-------------------------------------------------
0081  014D          PROC_NAIK:
0082  014D E5 90                   MOV     A,DATAKARTU
0083  014F 54 0F                   ANL     A,#0FH
0084  0151 B4 00 01                CJNE    A,#00H,CEKKARTU_N
0085  0154 22                      RET
0086  0155
0087  0155 12 03 55    CEKKARTU_N:    LCALL   DELAY
0088  0158 12 03 55                LCALL   DELAY
0089  015B E5 90                   MOV     A,DATAKARTU
0090  015D 54 0F                   ANL     A,#0FH
0091  015F B4 00 01                CJNE    A,#00H,KARTU_N1
0092  0162 22                      RET
0093  0163
0094  0163 B4 01 1E    KARTU_N1:      CJNE    A,#01H,KARTU_N2
0095  0166 75 30 00                MOV     B_PENUMPANG1_1,#0
0096  0169 75 31 00                MOV     B_PENUMPANG1_2,#0
0097  016C E5 42                   MOV     A,FLAGPENUMPANG1
0098  016E 54 FE                   ANL     A,#0FEH
0099  0170 44 01                   ORL     A,#01H
0100  0172 F5 42                   MOV     FLAGPENUMPANG1,A
0101  0174 75 38 BF                MOV     BUFSEG1,#0BFH
0102  0177 75 39 BF                MOV     BUFSEG2,#0BFH
0103  017A 75 3A BF                MOV     BUFSEG3,#0BFH
0104  017D 75 3B F9                MOV     BUFSEG4,#0F9H
```

```
0105  0180 12 02 64              LCALL   SCANTAMPILAN
0106  0183 22                   RET
0107  0184 B4 02 1E   KARTU_N2:     CJNE    A,#02H,KARTU_N3
0108  0187 75 32 00              MOV     B_PENUMPANG2_1,#0
0109  018A 75 33 00              MOV     B_PENUMPANG2_2,#0
0110  018D E5 42                MOV     A,FLAGPENUMPANG1
0111  018F 54 FD                ANL     A,#0FDH
0112  0191 44 02                ORL     A,#02H
0113  0193 F5 42                MOV     FLAGPENUMPANG1,A
0114  0195 75 38 BF              MOV     BUFSEG1,#0BFH
0115  0198 75 39 BF              MOV     BUFSEG2,#0BFH
0116  019B 75 3A BF              MOV     BUFSEG3,#0BFH
0117  019E 75 3B A4              MOV     BUFSEG4,#0A4H
0118  01A1 12 02 64              LCALL   SCANTAMPILAN
0119  01A4 22                   RET
0120  01A5 B4 03 1E   KARTU_N3:     CJNE    A,#03H,KARTU_N4
0121  01A8 75 34 00              MOV     B_PENUMPANG3_1,#0
0122  01AB 75 35 00              MOV     B_PENUMPANG3_2,#0
0123  01AE E5 42                MOV     A,FLAGPENUMPANG1
0124  01B0 54 FB                ANL     A,#0FBH
0125  01B2 44 04                ORL     A,#04H
0126  01B4 F5 42                MOV     FLAGPENUMPANG1,A
0127  01B6 75 38 BF              MOV     BUFSEG1,#0BFH
0128  01B9 75 39 BF              MOV     BUFSEG2,#0BFH
0129  01BC 75 3A BF              MOV     BUFSEG3,#0BFH
0130  01BF 75 3B B0              MOV     BUFSEG4,#0B0H
0131  01C2 12 02 64              LCALL   SCANTAMPILAN
0132  01C5 22                   RET
0133  01C6 B4 04 1E   KARTU_N4:     CJNE    A,#04H,KARTU_NN
0134  01C9 75 36 00              MOV     B_PENUMPANG4_1,#0
0135  01CC 75 37 00              MOV     B_PENUMPANG4_2,#0
0136  01CF E5 42                MOV     A,FLAGPENUMPANG1
0137  01D1 54 F7                ANL     A,#0F7H
0138  01D3 44 08                ORL     A,#08H
0139  01D5 F5 42                MOV     FLAGPENUMPANG1,A
0140  01D7 75 38 BF              MOV     BUFSEG1,#0BFH
0141  01DA 75 39 BF              MOV     BUFSEG2,#0BFH
0142  01DD 75 3A BF              MOV     BUFSEG3,#0BFH
0143  01E0 75 3B 99              MOV     BUFSEG4,#99H
0144  01E3 12 02 64              LCALL   SCANTAMPILAN
0145  01E6 22                   RET
0146  01E7          KARTU_NN:
0147  01E7 22                   RET
0148  01E8
0149  01E8          ;-------------------------------------------------
0150  01E8          ; PROCEDURE TURUN
0151  01E8          ;-------------------------------------------------
0152  01E8          PROC_TURUN:
0153  01E8 E5 90                MOV     A,DATAKARTU
0154  01EA C4                   SWAP    A
0155  01EB 54 0F                ANL     A,#0FH
0156  01ED B4 00 01              CJNE    A,#00H,CEKKARTU_T
0157  01F0 22                   RET
0158  01F1
0159  01F1 12 03 55   CEKKARTU_T:    LCALL   DELAY
0160  01F4 12 03 55              LCALL   DELAY
0161  01F7 E5 90                MOV     A,DATAKARTU
0162  01F9 C4                   SWAP    A
0163  01FA 54 0F                ANL     A,#0FH
0164  01FC B4 00 01              CJNE    A,#00H,KARTU_T1
0165  01FF 22                   RET
0166  0200
0167  0200 B4 01 0F   KARTU_T1:     CJNE    A,#01H,KARTU_T2
0168  0203 85 30 3E              MOV     BIAYA1,B_PENUMPANG1_1
0169  0206 85 31 3F              MOV     BIAYA2,B_PENUMPANG1_2
0170  0209 E5 42                MOV     A,FLAGPENUMPANG1
0171  020B 54 FE                ANL     A,#0FEH
0172  020D F5 42                MOV     FLAGPENUMPANG1,A
0173  020F 02 02 48              LJMP    TAMPILKANBIAYA
```

```
0174  0212 B4 02 0F   KARTU_T2:      CJNE    A,#02H,KARTU_T3
0175  0215 85 32 3E                  MOV     BIAYA1,B_PENUMPANG2_1
0176  0218 85 33 3F                  MOV     BIAYA2,B_PENUMPANG2_2
0177  021B E5 42                    MOV     A,FLAGPENUMPANG1
0178  021D 54 FD                    ANL     A,#0FDH
0179  021F F5 42                    MOV     FLAGPENUMPANG1,A
0180  0221 02 02 48                 LJMP    TAMPILKANBIAYA
0181  0224 B4 03 0F   KARTU_T3:      CJNE    A,#03H,KARTU_T4
0182  0227 85 34 3E                  MOV     BIAYA1,B_PENUMPANG3_1
0183  022A 85 35 3F                  MOV     BIAYA2,B_PENUMPANG3_2
0184  022D E5 42                    MOV     A,FLAGPENUMPANG1
0185  022F 54 FB                    ANL     A,#0FBH
0186  0231 F5 42                    MOV     FLAGPENUMPANG1,A
0187  0233 02 02 48                 LJMP    TAMPILKANBIAYA
0188  0236 B4 04 32   KARTU_T4:      CJNE    A,#04H,KARTU_TN
0189  0239 85 36 3E                  MOV     BIAYA1,B_PENUMPANG4_1
0190  023C 85 37 3F                  MOV     BIAYA2,B_PENUMPANG4_2
0191  023F E5 42                    MOV     A,FLAGPENUMPANG1
0192  0241 54 F7                    ANL     A,#0F7H
0193  0243 F5 42                    MOV     FLAGPENUMPANG1,A
0194  0245 02 02 48                 LJMP    TAMPILKANBIAYA
0195  0248
0196  0248 E5 3E      TAMPILKANBIAYA: MOV     A,BIAYA1
0197  024A 12 02 72                  LCALL   ANDF0
0198  024D F5 38                    MOV     BUFSEG1,A
0199  024F E5 3E                    MOV     A,BIAYA1
0200  0251 12 02 6C                  LCALL   AND0F
0201  0254 F5 39                    MOV     BUFSEG2,A
0202  0256 E5 3F                    MOV     A,BIAYA2
0203  0258 12 02 72                  LCALL   ANDF0
0204  025B F5 3A                    MOV     BUFSEG3,A
0205  025D E5 3F                    MOV     A,BIAYA2
0206  025F 12 02 6C                  LCALL   AND0F
0207  0262 F5 3B                    MOV     BUFSEG4,A
0208  0264          SCANTAMPILAN:
0209  0264 7B 8F                    MOV     R3,#$8F
0210  0266          TAMPILTERUS:
0211  0266 12 02 E6                  LCALL   SCANNING
0212  0269 DB FB                    DJNZ    R3,TAMPILTERUS
0213  026B
0214  026B          KARTU_TN:
0215  026B 22                      RET
0216  026C          ;-----------------------------------------
0217  026C
0218  026C 54 0F      AND0F:        ANL     A,#0FH
0219  026E 12 03 2D                  LCALL   CONV7SEG
0220  0271 22                      RET
0221  0272 54 F0      ANDF0:        ANL     A,#0F0H
0222  0274 C4                      SWAP    A
0223  0275 12 03 2D                  LCALL   CONV7SEG
0224  0278 22                      RET
0225  0279
0226  0279
0227  0279
0228  0279 E5 42      PROC_HITUNG:    MOV     A,FLAGPENUMPANG1
0229  027B 54 0F                    ANL     A,#$0F
0230  027D B4 00 01                  CJNE    A,#0,CEKFLAG1
0231  0280 22                      RET
0232  0281 E5 42      CEKFLAG1:     MOV     A,FLAGPENUMPANG1
0233  0283 54 01                    ANL     A,#01H
0234  0285 B4 01 03                  CJNE    A,#01H,CEKFLAG2
0235  0288 12 02 AA                  LCALL   HITUNG_P1
0236  028B E5 42      CEKFLAG2:     MOV     A,FLAGPENUMPANG1
0237  028D 54 02                    ANL     A,#02H
0238  028F B4 02 03                  CJNE    A,#02,CEKFLAG3
0239  0292 12 02 B9                  LCALL   HITUNG_P2
0240  0295 E5 42      CEKFLAG3:     MOV     A,FLAGPENUMPANG1
0241  0297 54 04                    ANL     A,#04H
0242  0299 B4 04 03                  CJNE    A,#04H,CEKFLAG4
```

```
0243  029C 12 02 C8              LCALL   HITUNG_P3
0244  029F E5 42      CEKFLAG4:   MOV     A,FLAGPENUMPANG1
0245  02A1 54 08                 ANL     A,#08H
0246  02A3 B4 08 03               CJNE    A,#08H,CEKFLAGERR
0247  02A6 12 02 D7               LCALL   HITUNG_P4
0248  02A9 22        CEKFLAGERR:  RET
0249  02AA
0250  02AA
0251  02AA          HITUNG_P1:
0252  02AA          ;----BIAYA
0253  02AA E5 31                  MOV     A,B_PENUMPANG1_2
0254  02AC 24 08                  ADD     A,#BSATUMETER2
0255  02AE D4                     DA      A
0256  02AF F5 31                  MOV     B_PENUMPANG1_2,A
0257  02B1 E5 30                  MOV     A,B_PENUMPANG1_1
0258  02B3 34 00                  ADDC    A,#BSATUMETER1
0259  02B5 D4                     DA      A
0260  02B6 F5 30                  MOV     B_PENUMPANG1_1,A
0261  02B8 22                     RET
0262  02B9
0263  02B9          HITUNG_P2:
0264  02B9          ;----BIAYA
0265  02B9 E5 33                  MOV     A,B_PENUMPANG2_2
0266  02BB 24 08                  ADD     A,#BSATUMETER2
0267  02BD D4                     DA      A
0268  02BE F5 33                  MOV     B_PENUMPANG2_2,A
0269  02C0 E5 32                  MOV     A,B_PENUMPANG2_1
0270  02C2 34 00                  ADDC    A,#BSATUMETER1
0271  02C4 D4                     DA      A
0272  02C5 F5 32                  MOV     B_PENUMPANG2_1,A
0273  02C7
0274  02C7 22                     RET
0275  02C8          HITUNG_P3:
0276  02C8          ;----BIAYA
0277  02C8 E5 35                  MOV     A,B_PENUMPANG3_2
0278  02CA 24 08                  ADD     A,#BSATUMETER2
0279  02CC D4                     DA      A
0280  02CD F5 35                  MOV     B_PENUMPANG3_2,A
0281  02CF E5 34                  MOV     A,B_PENUMPANG3_1
0282  02D1 34 00                  ADDC    A,#BSATUMETER1
0283  02D3 D4                     DA      A
0284  02D4 F5 34                  MOV     B_PENUMPANG3_1,A
0285  02D6
0286  02D6 22                     RET
0287  02D7          HITUNG_P4:
0288  02D7          ;----BIAYA
0289  02D7 E5 37                  MOV     A,B_PENUMPANG4_2
0290  02D9 24 08                  ADD     A,#BSATUMETER2
0291  02DB D4                     DA      A
0292  02DC F5 37                  MOV     B_PENUMPANG4_2,A
0293  02DE E5 36                  MOV     A,B_PENUMPANG4_1
0294  02E0 34 00                  ADDC    A,#BSATUMETER1
0295  02E2 D4                     DA      A
0296  02E3 F5 36                  MOV     B_PENUMPANG4_1,A
0297  02E5 22                     RET
0298  02E6
0299  02E6
0300  02E6          ;-------------------------------------------------
0301  02E6          SCANNING:
0302  02E6 85 38 80  SCAN1:       MOV     DATA7SEG,BUFSEG1
0303  02E9 C2 A0                  CLR     SLCT7SEG1
0304  02EB D2 A1                  SETB    SLCT7SEG2
0305  02ED D2 A2                  SETB    SLCT7SEG3
0306  02EF D2 A3                  SETB    SLCT7SEG4
0307  02F1 12 03 1E               LCALL   CLEARKONTROL
0308  02F4
0309  02F4 85 39 80  SCAN2:       MOV     DATA7SEG,BUFSEG2
0310  02F7 D2 A0                  SETB    SLCT7SEG1
0311  02F9 C2 A1                  CLR     SLCT7SEG2
```

```
0312  02FB D2 A2                SETB    SLCT7SEG3
0313  02FD D2 A3                SETB    SLCT7SEG4
0314  02FF 12 03 1E             LCALL   CLEARKONTROL
0315  0302
0316  0302 85 3A 80   SCAN3:     MOV     DATA7SEG,BUFSEG3
0317  0305 D2 A0                SETB    SLCT7SEG1
0318  0307 D2 A1                SETB    SLCT7SEG2
0319  0309 C2 A2                CLR     SLCT7SEG3
0320  030B D2 A3                SETB    SLCT7SEG4
0321  030D 12 03 1E             LCALL   CLEARKONTROL
0322  0310
0323  0310 85 3B 80   SCAN4:     MOV     DATA7SEG,BUFSEG4
0324  0313 D2 A0                SETB    SLCT7SEG1
0325  0315 D2 A1                SETB    SLCT7SEG2
0326  0317 D2 A2                SETB    SLCT7SEG3
0327  0319 C2 A3                CLR     SLCT7SEG4
0328  031B 12 03 1E             LCALL   CLEARKONTROL
0329  031E
0330  031E
0331  031E 12 03 47   CLEARKONTROL:  LCALL   DELAYDISPON
0332  0321 D2 A0                SETB    SLCT7SEG1
0333  0323 D2 A1                SETB    SLCT7SEG2
0334  0325 D2 A2                SETB    SLCT7SEG3
0335  0327 D2 A3                SETB    SLCT7SEG4
0336  0329 12 03 50             LCALL   DELAYDISP
0337  032C 22                   RET
0338  032D
0339  032D
0340  032D
0341  032D
0342  032D
0343  032D          ;------ ROUTINE KONVERSI ANGKA KE DALAM FORMAT 7 SEGMENT ----
0344  032D          CONV7SEG:
0345  032D
0346  032D FE                   MOV     R6,A
0347  032E 90 03 5E             MOV     DPTR,#SEG
0348  0331 E4                   CLR     A
0349  0332 BE 00 03             CJNE    R6,#00,INCDPTR
0350  0335 02 03 3B             LJMP    LOAD
0351  0338 A3       INCDPTR:    INC     DPTR
0352  0339 DE FD                DJNZ    R6,INCDPTR
0353  033B 93       LOAD:       MOVC    A,@A+DPTR
0354  033C 22                   RET
0355  033D
0356  033D          PROC_HAPUSSEMUABUFFER:
0357  033D 78 30                MOV     R0,#B_PENUMPANG1_1
0358  033F 7C 38                MOV     R4,#56
0359  0341          HAPUSSEMUABUFFER:
0360  0341 76 00                MOV     @R0,#0
0361  0343 08                   INC     R0
0362  0344 DC FB                DJNZ    R4,HAPUSSEMUABUFFER
0363  0346 22                   RET
0364  0347
0365  0347          DELAYDISPON:
0366  0347 7E 08                MOV     R6,#$08
0367  0349 7F FF    DELAY0ON:   MOV     R7,#$FF
0368  034B DF FE    DELAY1ON:   DJNZ    R7,DELAY1ON
0369  034D DE FA                DJNZ    R6,DELAY0ON
0370  034F 22                   RET
0371  0350
0372  0350          DELAYDISP:
0373  0350          ;           MOV     R6,#$01
0374  0350 7F 2F    DELAY0:     MOV     R7,#$2F
0375  0352 DF FE    DELAY1:     DJNZ    R7,DELAY1
0376  0354          ;           DJNZ    R6,DELAY0
0377  0354 22                   RET
0378  0355
0379  0355          DELAY:
0380  0355 7E FF                MOV     R6,#$FF
```

```
0381   0357 7F FF      DELAYL0:        MOV     R7,#$FF
0382   0359 DF FE      DELAYL1:        DJNZ    R7,DELAYL1
0383   035B DE FA                      DJNZ    R6,DELAYL0
0384   035D 22                         RET
0385   035E            ;---------------------------------------------------------
0386   035E            ;  LOOKUP TABLE
0387   035E            ;---------------------------------------------------------
0388   035E C0F9A4B09992SEG:           .BYTE
0C0H,0F9H,0A4H,0B0H,99H,92H,82H,0F8H,80H,90H,0C0H,0BFH,0BFH
0388   0364 82F88090C0BFBF
0389   036B
0390   036B                            .END
tasm: Number of errors = 0
```

LAMPIRAN D

DATA KOMPONEN

SIDANG TUGAS AKHIR

Perancangan Tarif Biaya Angkot dengan Mikrokontroler AT89C51

Nama : I Wayan Sunarto
NRP : 9622102
Pembimbing TA : Marvin Chandra Wijaya,ST.,MM.,MT

Latar Belakang

- Perselisihan sering terjadi antara penumpang dengan supir angkot yang disebabkan oleh biaya.
- Beda penafsiran tarif biaya

Identifikasi Masalah

Bagaimana merancang & merealisasikan suatu alat pencatat tarif biaya angkot berdasarkan jarak tempuh?

Tujuan

Maksud dan tujuan tugas akhir ini adalah merancang dan merealisasikan suatu alat pencatat tarif biaya angkot berdasarkan jarak tempuh.

Pembatasan Masalah dan Spesifikasi Alat

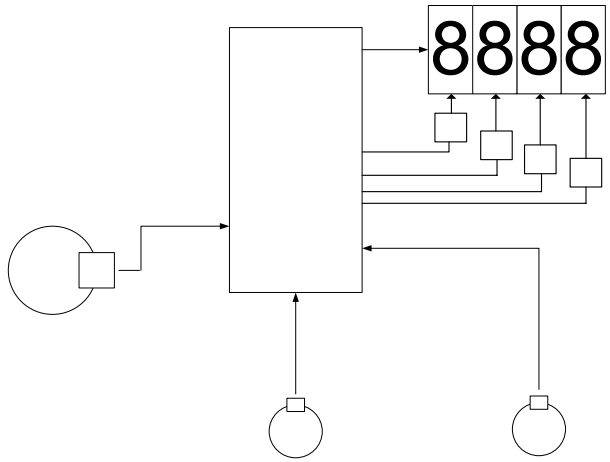
Pembatasan Masalah

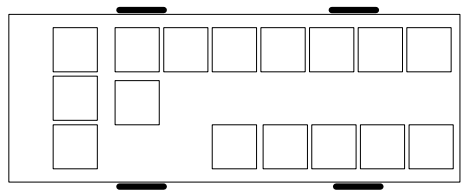
1. Input maksimal 15 penumpang
2. Tampilan output biaya maksimal 4 digit
3. Sensor kartu dan sensor jarak bekerja baik

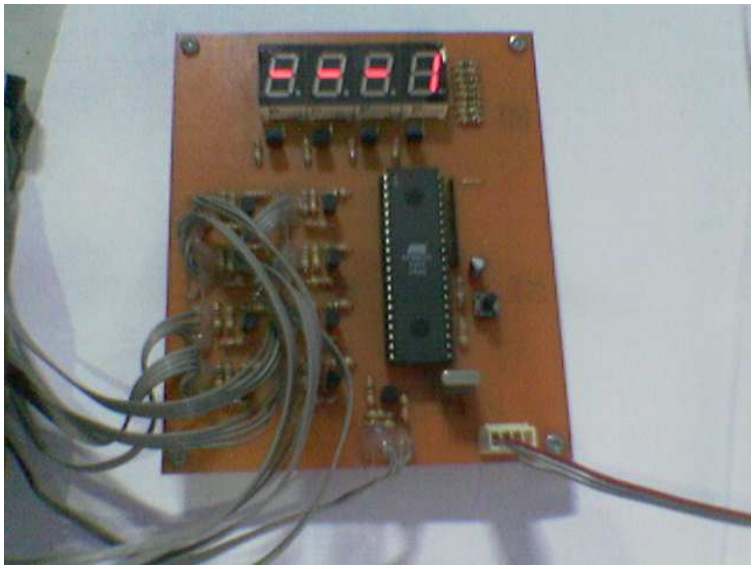
Spesifikasi Alat

1. Input 15 penumpang
2. Output 4 digit
3. Sensor optocoupler

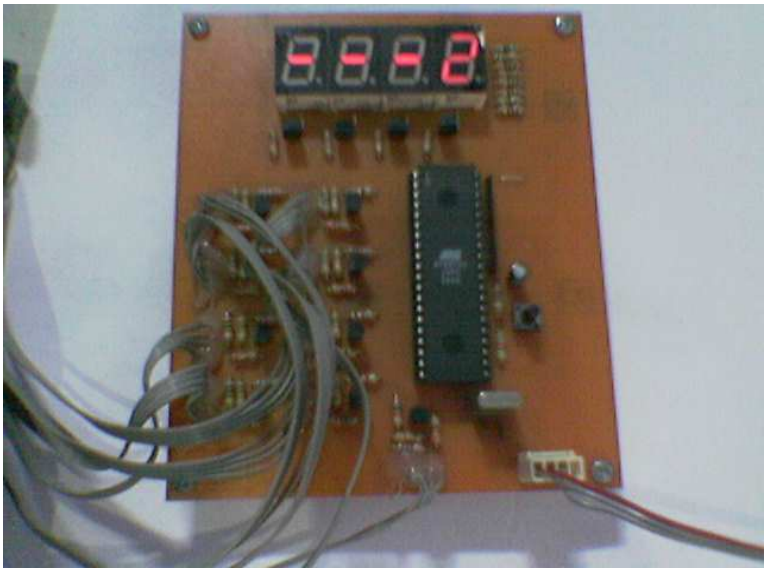
Blok Diagram



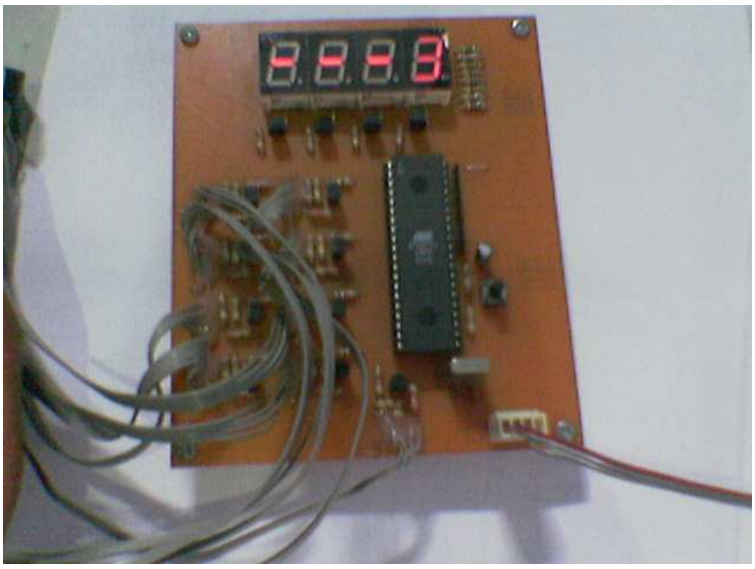




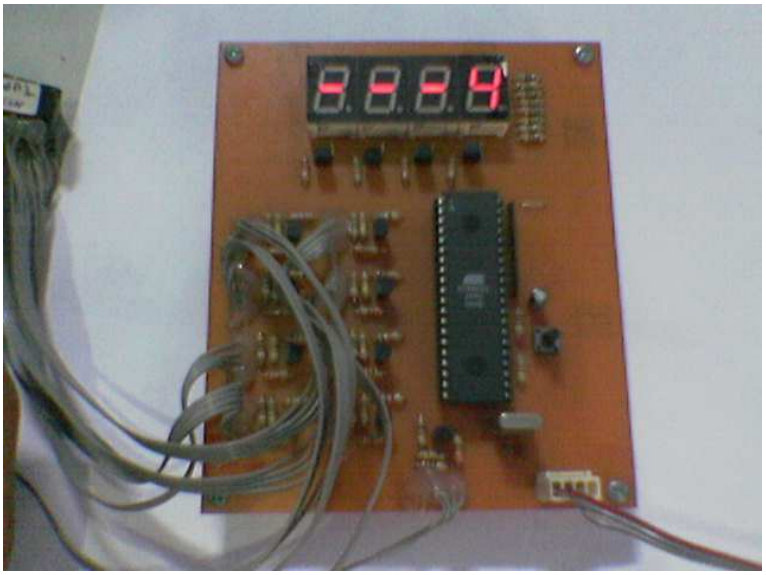
Gambar 3 Tampilan saat kartu penumpang 1 dimasukkan ke rangkaian naik



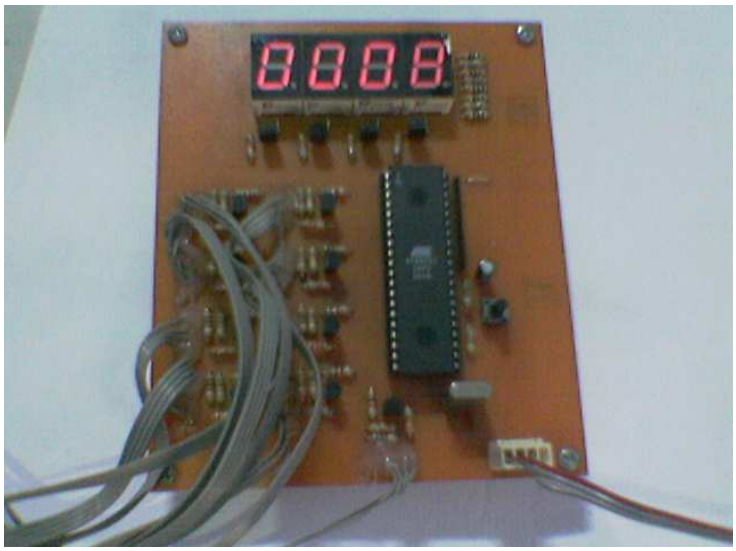
Gambar 4 Tampilan saat kartu penumpang 2 dimasukkan ke rangkaian naik



Gambar 5 Tampilan saat kartu penumpang 3 dimasukkan ke rangkaian naik

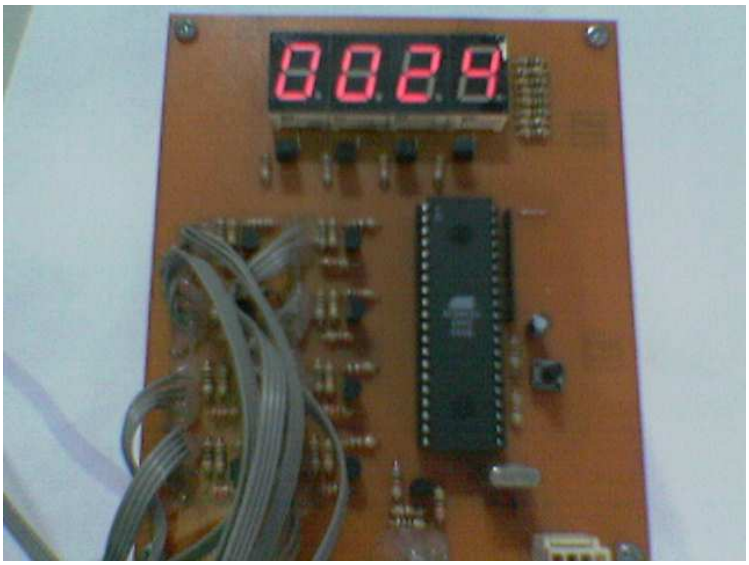


Gambar 6 Tampilan saat kartu penumpang 4 dimasukkan ke rangkaian naik

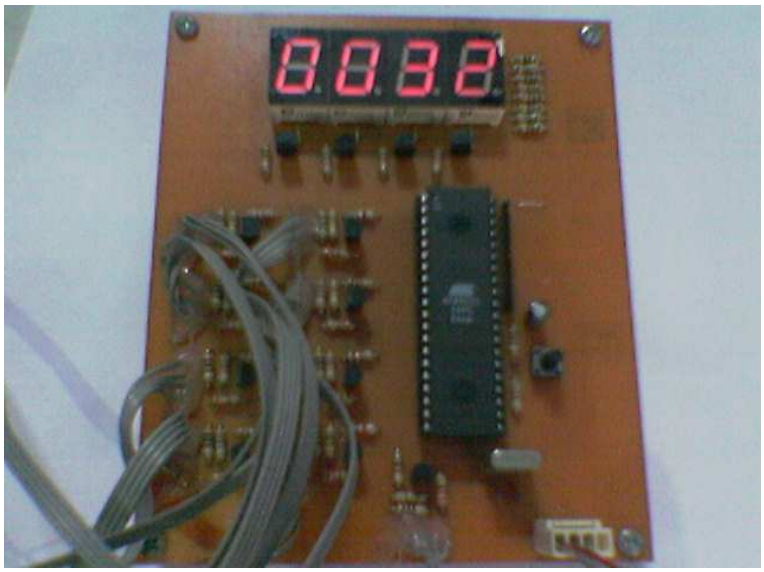


Gambar 7 Tampilan saat kartu penumpang 1 dimasukkan ke rangkaian turun dengan jarak tempuh 2 kedipan (1 meter)

Gambar 8 Tampilan saat kartu penumpang 2 dimasukkan ke rangkaian turun dengan jarak tempuh 4 kedipan (2 meter)

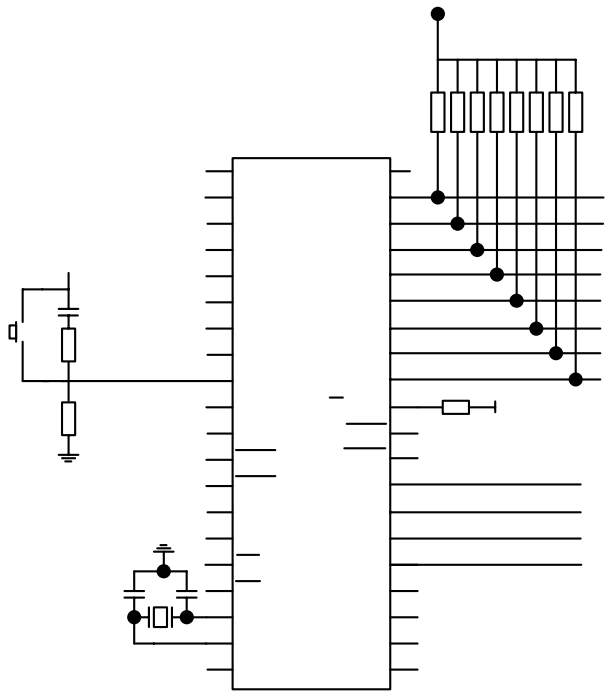


Gambar 9 Tampilan saat kartu penumpang 3 dimasukkan ke rangkaian turun dengan jarak tempuh 6 kedipan (3 meter)

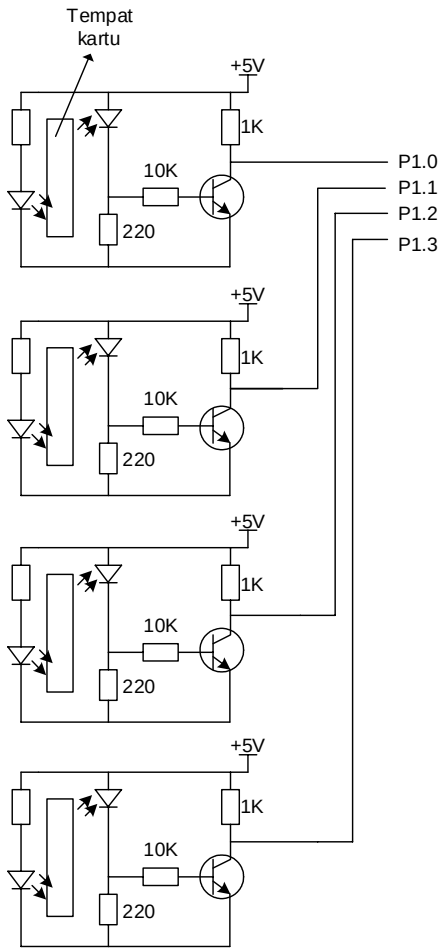


Gambar 10 Tampilan saat kartu penumpang 4 dimasukkan ke rangkaian turun dengan jarak tempuh 8 kedipan (4 meter)

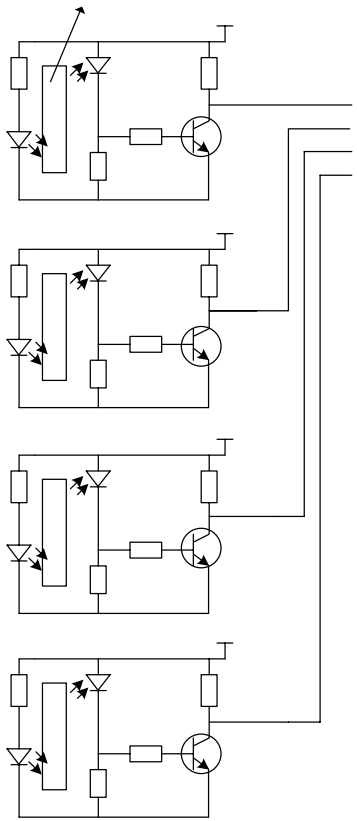
Rangkaian Mikrokontroler



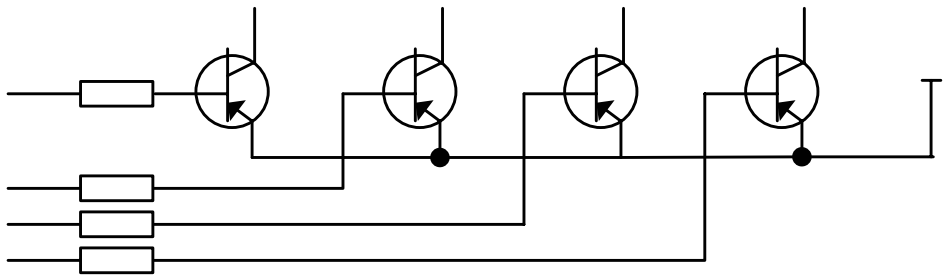
Rangkaian Naik



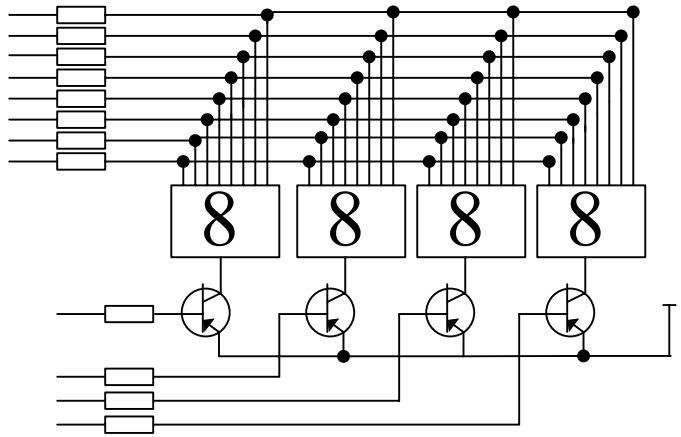
Rangkaian Turun



Rangkaian Scanning



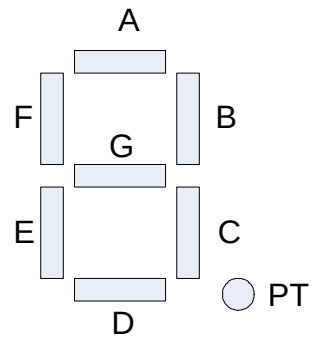
Rangkaian Seven Segment



Hubungan Port 0 Dengan Seven Segment

Hubungan Port 0 dengan Seven Segment

Mikrokontroler		Seven Segment (Common Anoda)
P0.0		PIN A
P0.1		PIN B
P0.2		PIN C
P0.3		PIN D
P0.4		PIN E
P0.5		PIN F
P0.6		PIN G
P0.7		PIN PT



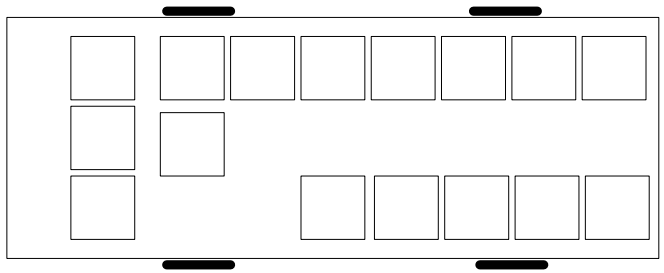
Seven Segment

Daftar Heksa Tampilan Seven Segment

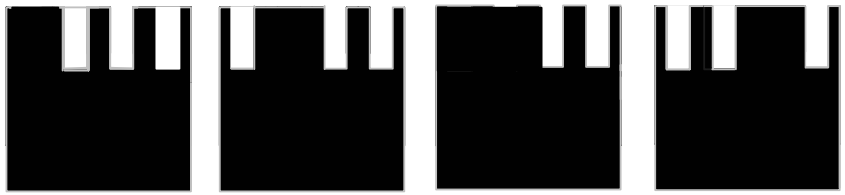
Daftar Heksa Tampilan Seven Segment

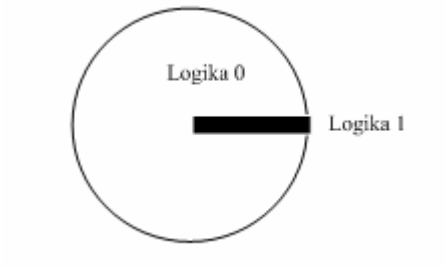
TAMPILAN ANGKA	P0.7 DP	P0.6 G	P0.5 F	P0.4 E	P0.3 D	P0.2 C	P0.1 B	P0.0 A	HEKSA DESIMAL
0	1	1	0	0	0	0	0	0	C0
1	1	0	1	1	1	0	0	1	F9
2	1	0	1	0	0	1	0	0	A4
3	1	0	1	1	0	0	0	0	B0
4	1	0	0	1	1	0	0	1	99
5	1	0	0	1	0	0	1	0	92
6	1	0	0	0	0	0	1	0	82
7	1	1	1	1	1	0	0	0	F8
8	1	0	0	0	0	0	0	0	80
9	1	0	0	1	0	0	0	0	90

Denah Tempat Duduk



Kartu Penumpang dan Kartu Piringan





Konversi Bilangan Kartu Penumpang

Konversi Bilangan Kartu Penumpang

Desimal	Heksadesimal	Biner	Kartu Penumpang
0	00	0000 0000	0
1	01	0000 0001	Kartu Penumpang 1
2	02	0000 0010	Kartu Penumpang 2
3	03	0000 0011	Kartu Penumpang 3
4	04	0000 0100	Kartu Penumpang 4
5	05	0000 0101	Kartu Penumpang 5
6	06	0000 0110	Kartu Penumpang 6
7	07	0000 0111	Kartu Penumpang 7
8	08	0000 1000	Kartu Penumpang 8
9	09	0000 1001	Kartu Penumpang 9
10	0A	0000 1010	Kartu Penumpang 10
11	0B	0000 1011	Kartu Penumpang 11
12	0C	0000 1100	Kartu Penumpang 12
13	0D	0000 1101	Kartu Penumpang 13
14	0E	0000 1110	Kartu Penumpang 14
15	0F	0000 1111	Kartu Penumpang 15

Diagram Alir Utama

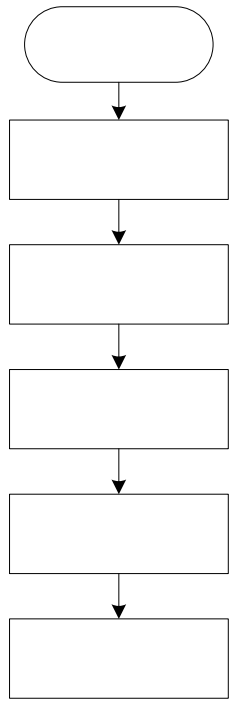


Diagram Alir Cek Kartu Naik

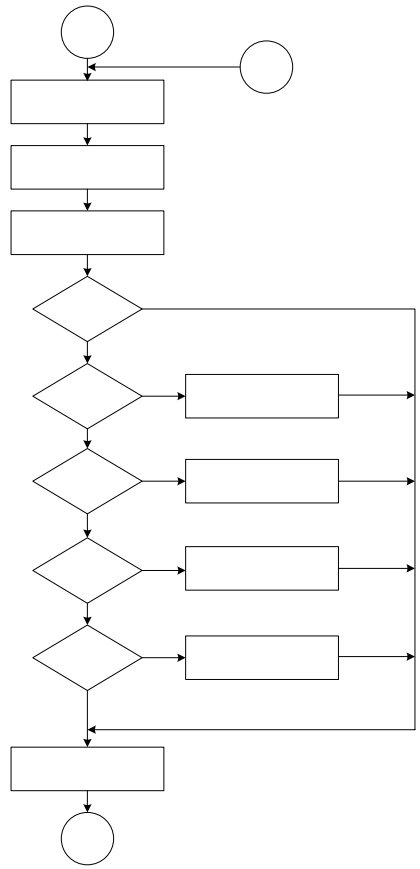


Diagram Alir Cek Kartu Turun

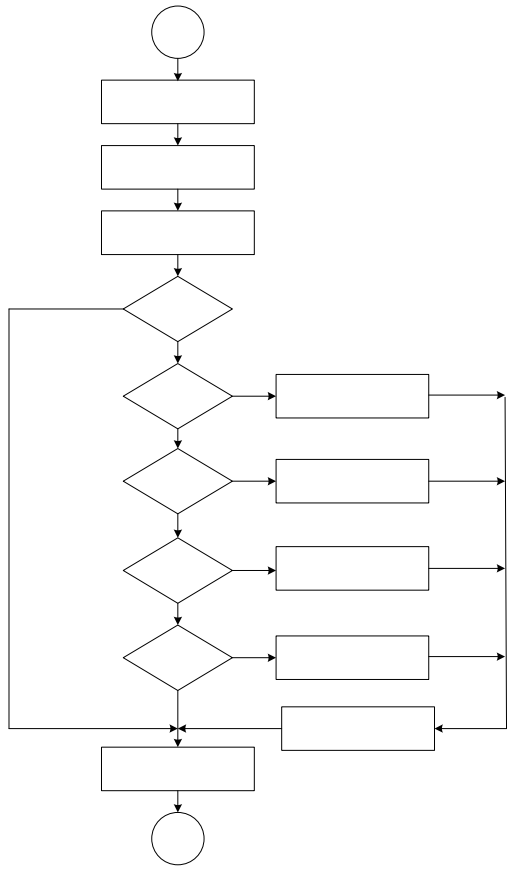


Diagram Alir Cek Kartu Piringan dan Tampilkan Biaya

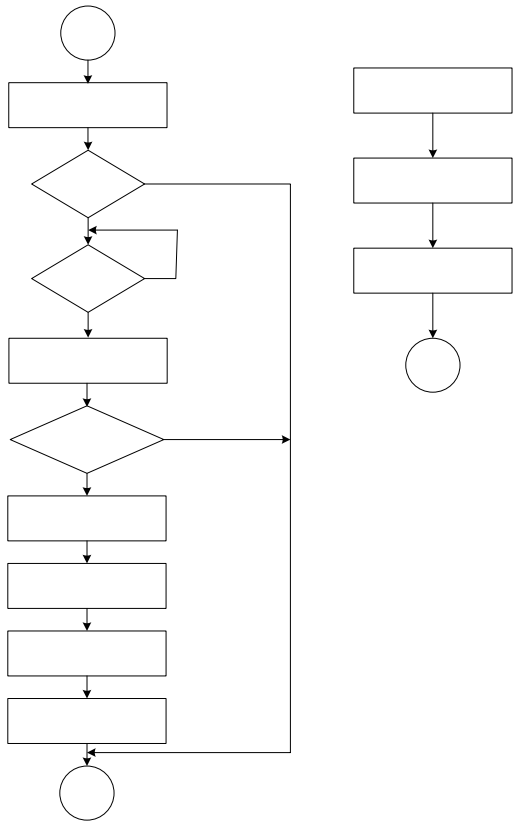
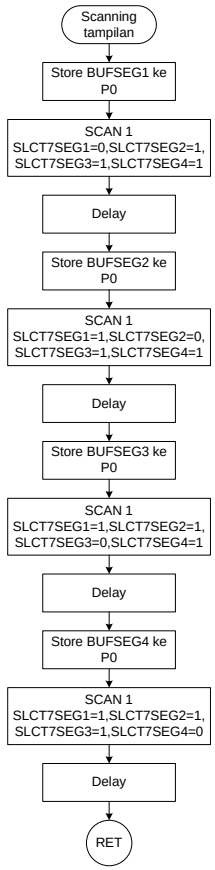


Diagram Alir Scanning Tampilan



Cara Kerja

- Sensor naik dan turun berfungsi untuk mendeteksi kartu penumpang
- Sensor jarak berfungsi untuk menghitung jarak tempuh penumpang
- Rangkaian scanning berfungsi untuk menghidupkan seven segment secara bergantian
- Seven segment berfunsi sebagai penampil output

Uji Coba Alat

Pengujian Rangkaian Naik, Jarak dan Turun

Pengujian kartu penumpang 1 sampai 4

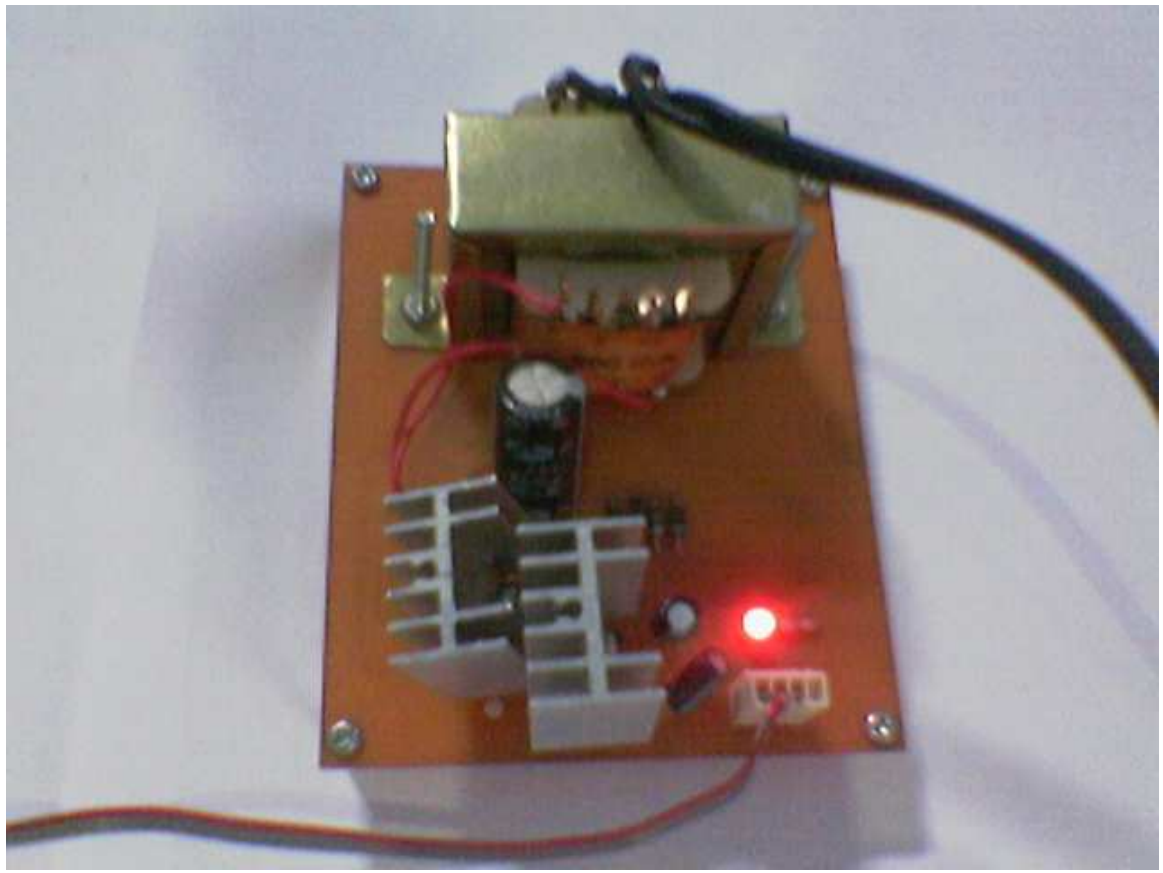
Kartu	Rangkaian Naik		Rangkaian Jarak		Rangkaian Turun
No	*Biner*	*Tamp. 7 Segment*	*Putaran*	*Meter*	*Tamp. 7 Segment*
1	0001	1	2(2 kedip)	1	8
2	0010	2	4(4 Kedip)	2	16
3	0011	3	6(6 kedip)	3	24
4	0100	4	8(8 kedip)	4	32
Ket: 2 putaran = 1 meter					
1 meter = Rp 8					

Pengujian Terhadap Seven Segment

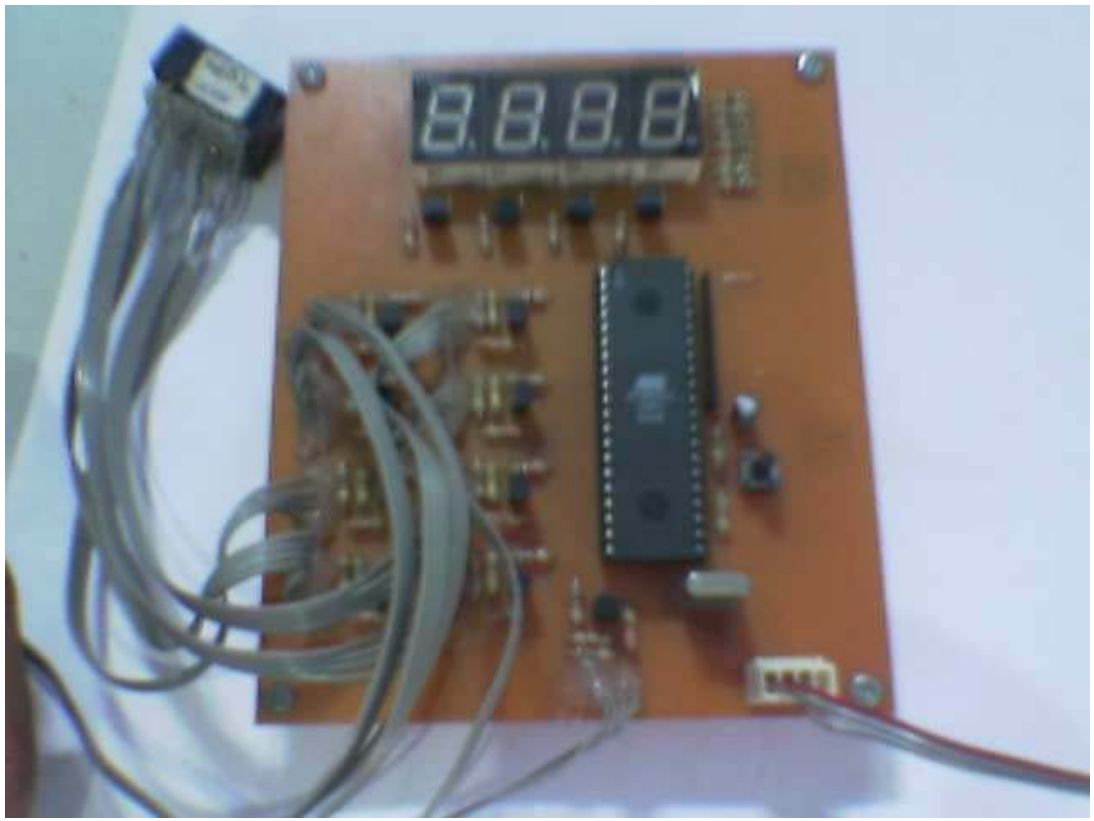
Tampilan Seven Segment

Sensor	Seven Segment	Ket
Sensor naik	Tampil flag penumpang	kartu naik di masukkan pada sensor naik
Sensor jarak	Tampil Kedip	Setiap satu putaran
Sensor turun	Tampil biaya	Untuk masing-masing kartu

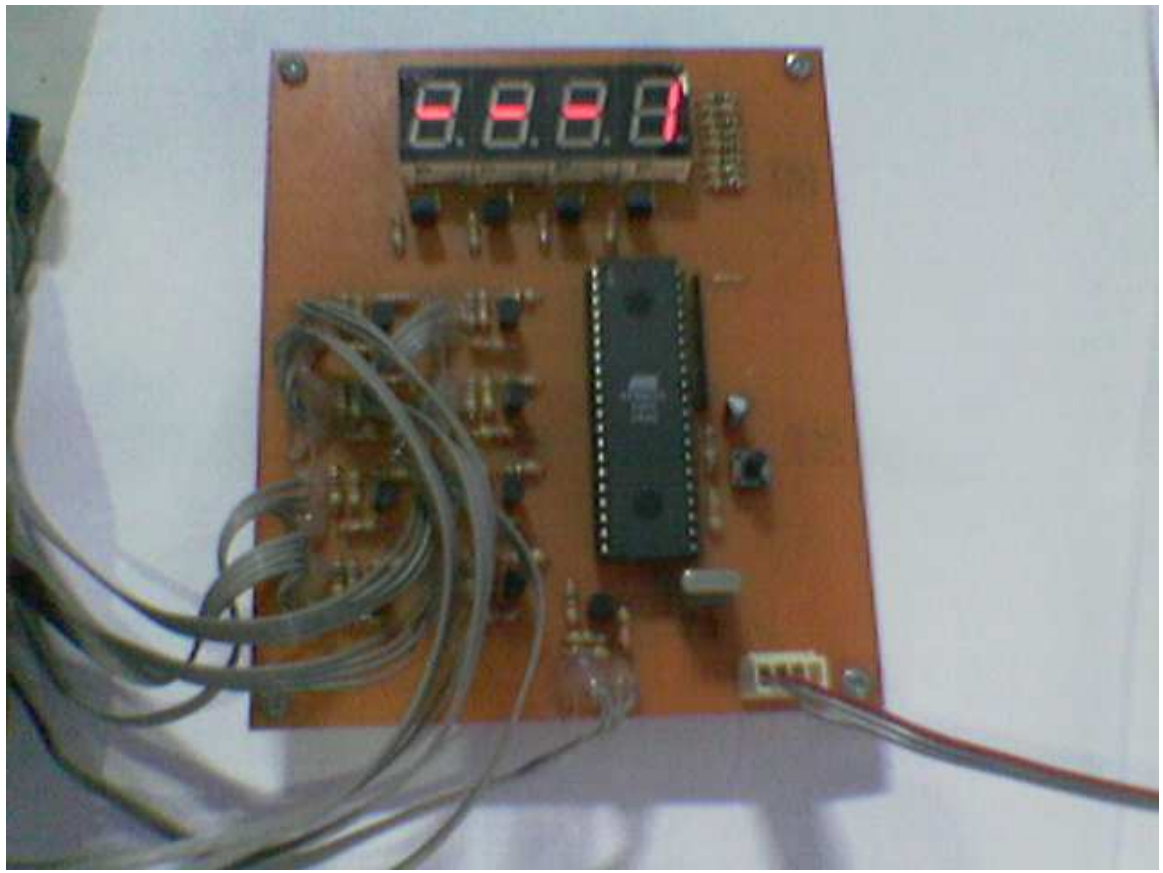
Regulator



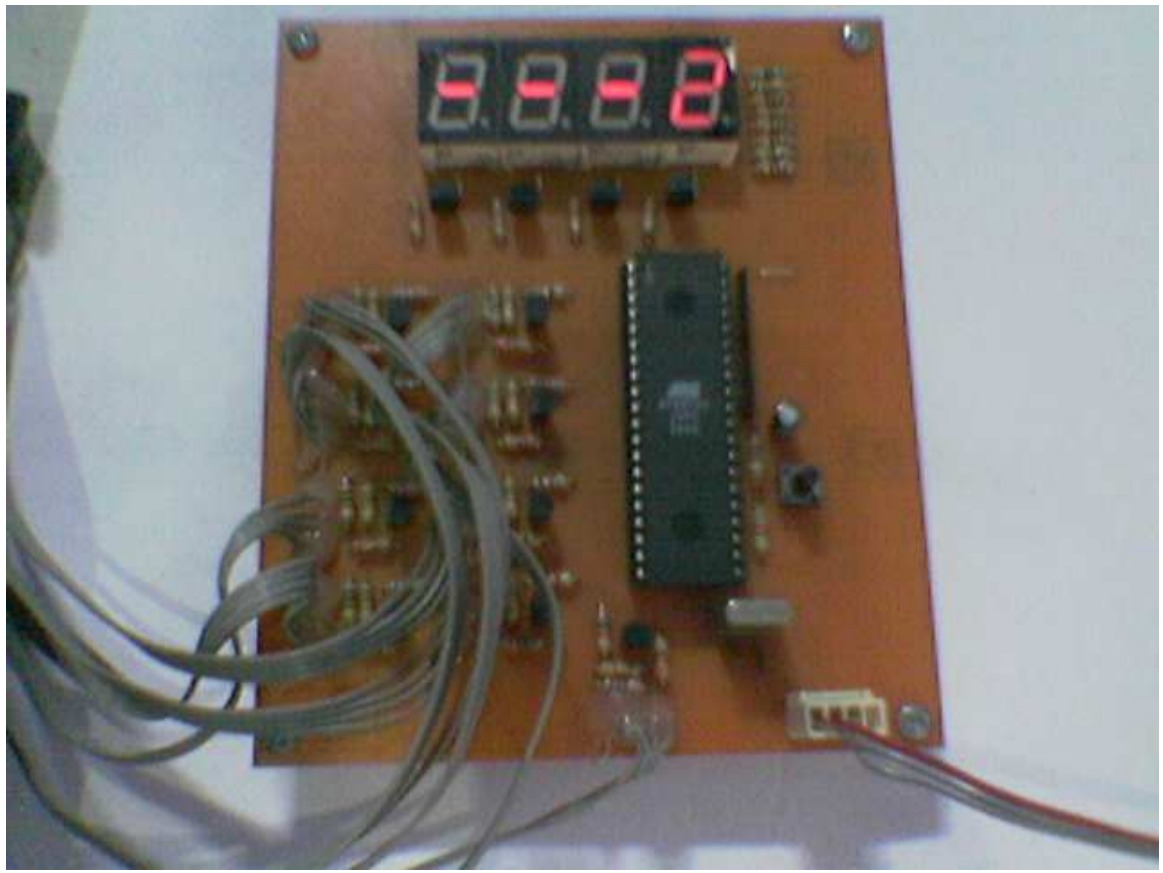
Alat Pencatat Tarif Biaya Angkot



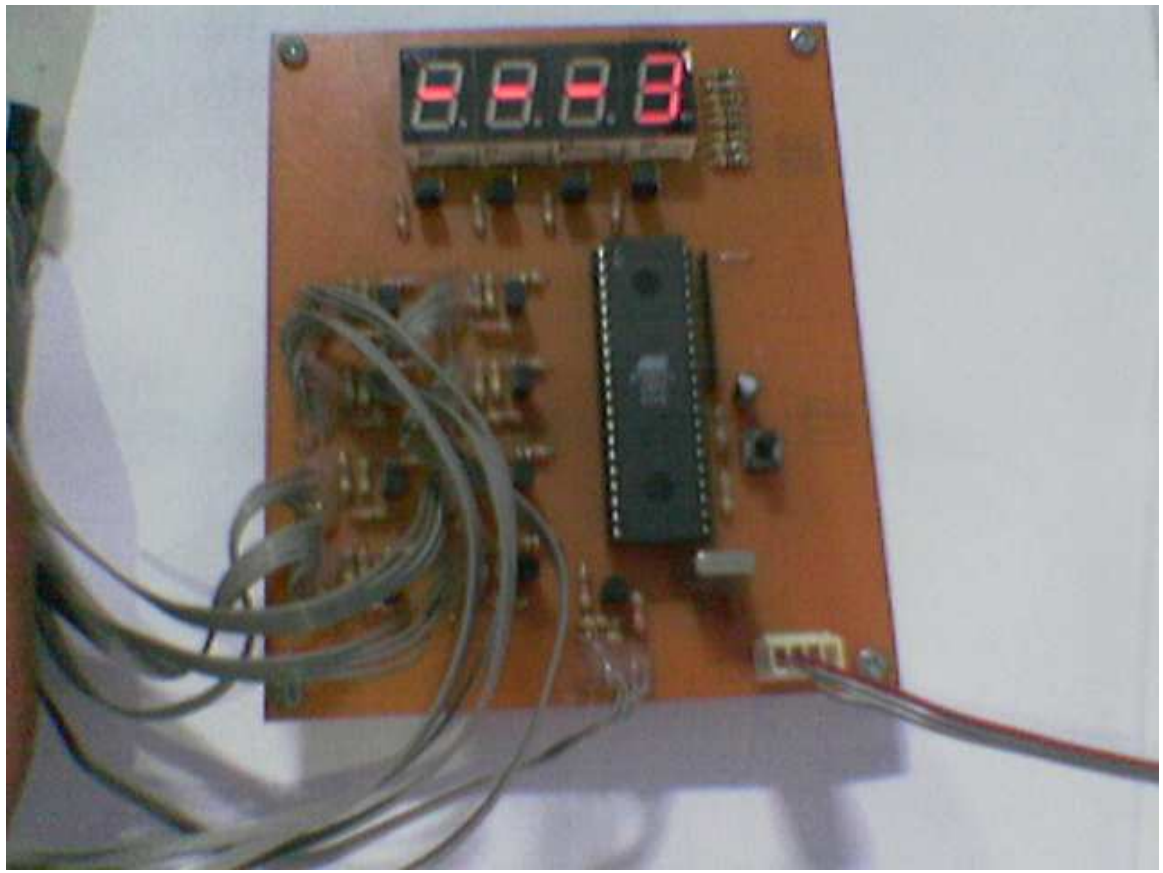
Tampilan saat kartu penumpang 1 dimasukan ke rangkaian naik



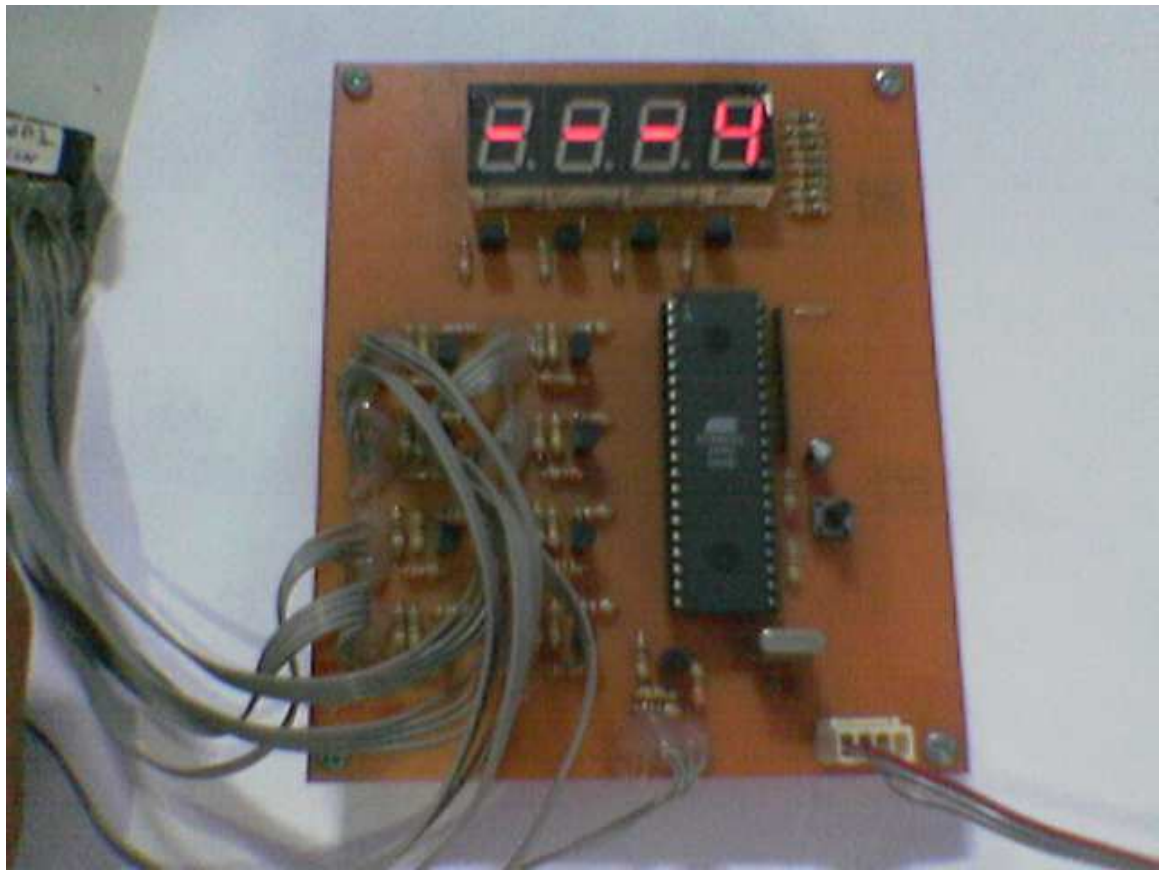
Tampilan saat kartu penumpang 2 dimasukan ke rangkaian naik



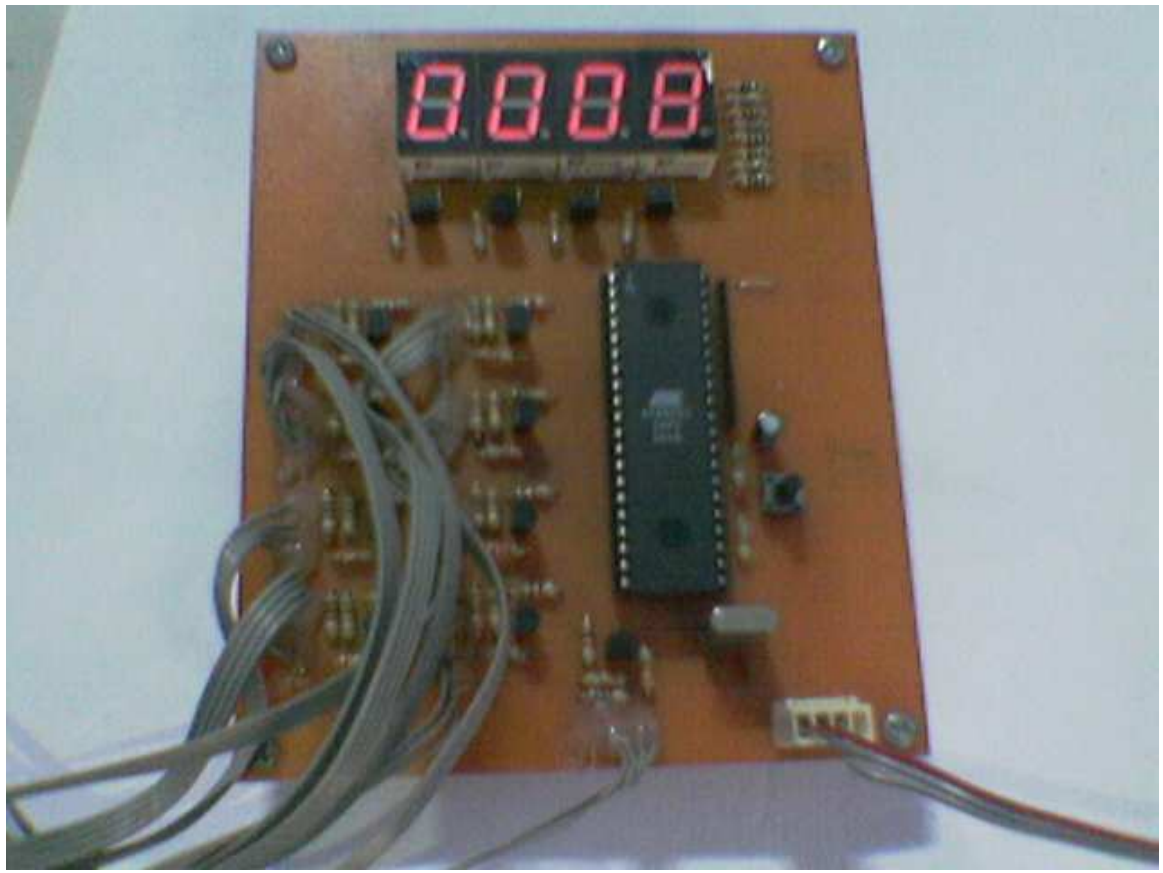
Tampilan saat kartu penumpang 3 dimasukan ke rangkaian naik



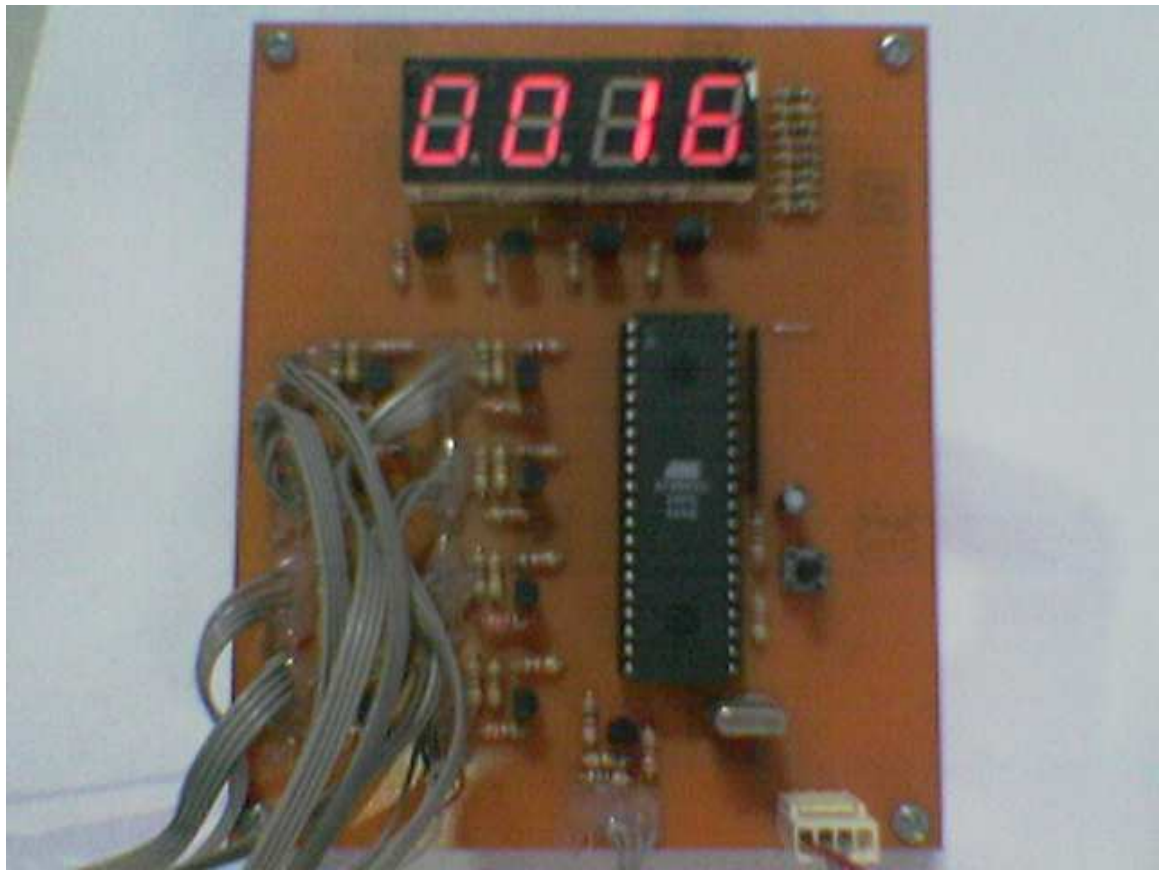
Tampilan saat kartu penumpang 4 dimasukan ke rangkaian naik



Tampilan saat kartu penumpang 1 dimasukan ke rangkaian turun dengan jarak tempuh 2 kedipan (1meter)

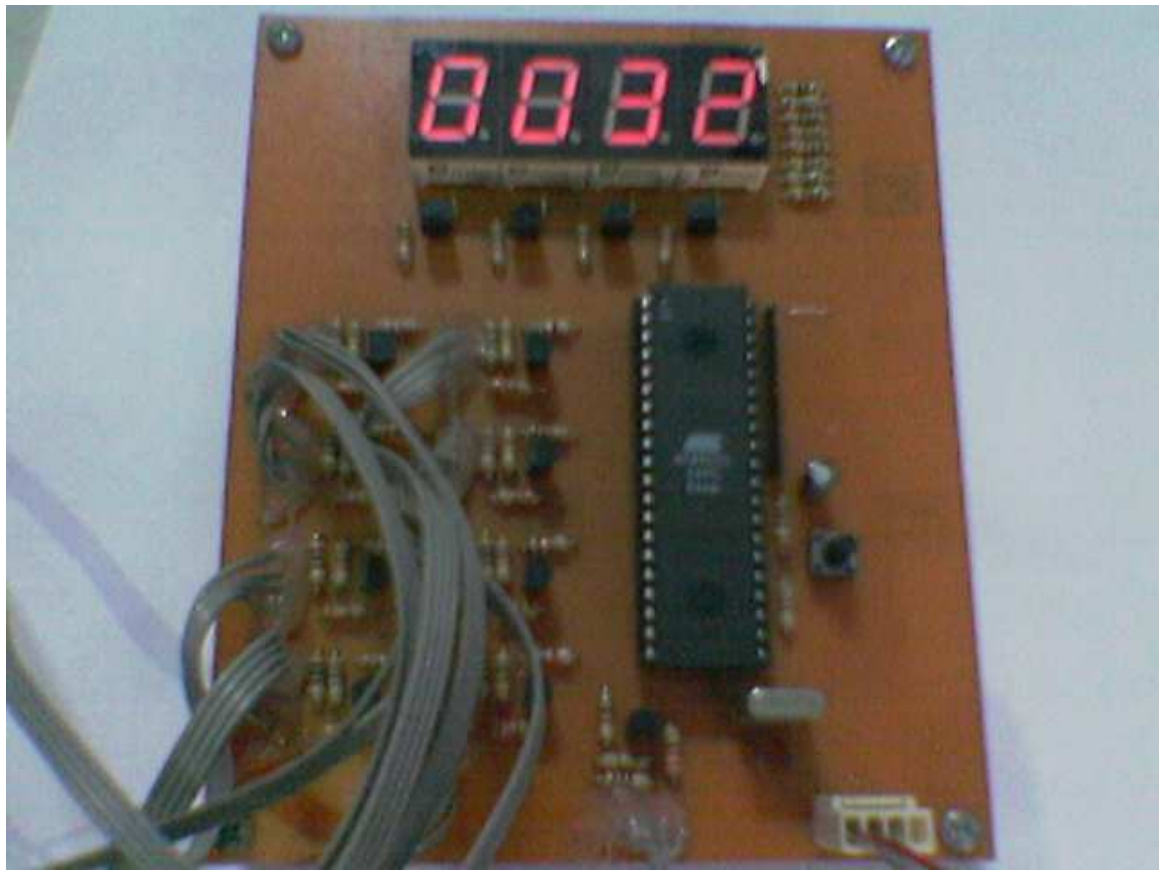


Tampilan saat kartu penumpang 2 dimasukan ke rangkaian turun dengan jarak tempuh 4 kedipan (2meter)



Tampilan saat kartu penumpang 3 dimasukan ke rangkaian turun dengan jarak tempuh 6 kedipan (3meter)

Tampilan saat kartu penumpang 4 dimasukan ke rangkaian turun dengan jarak tempuh 8 kedipan (4meter)



Kesimpulan dan Saran

Kesimpulan

- Hasil pengujian menunjukkan alat dapat digunakan sesuai dengan yang diharapkan.
- Penggunaan optocoupler masih mengalami kesulitan dalam mendeteksi kartu penumpang

Saran

- Untuk pengembangan lebih lanjut disarankan penggunaan optocoupler diganti dengan jenis yang lebih khusus untuk mempermudah dalam mendeteksi kartu