LAMPIRAN A
DATASHEET

Features

- High-performance, Low-power Atmel® AVR® 8-bit Microcontroller
- Advanced RISC Architecture
  - 131 Powerful Instructions – Most Single-clock Cycle Execution
  - 32 x 8 General Purpose Working Registers
  - Fully Static Operation
  - Up to 16 MIPS Throughput at 16 MHz
  - On-chip 2-cycle Multiplier
- High Endurance Non-volatile Memory segments
  - 16 Kbytes of In-System Self-programmable Flash program memory
  - 512 Bytes EEPROM
  - 1 Kbyte Internal SRAM
  - Write/Erase Cycles: 10,000 Flash/100,000 EEPROM
  - Data retention: 20 years at 85°C/100 years at 25°C(1)
  - Optional Boot Code Section with Independent Lock Bits
    In-System Programming by On-chip Boot Program
    True Read-While-Write Operation
  - Programming Lock for Software Security
- JTAG (IEEE std. 1149.1 Compliant) Interface
  - Boundary-scan Capabilities According to the JTAG Standard
  - Extensive On-chip Debug Support
  - Programming of Flash, EEPROM, Fuses, and Lock Bits through the JTAG Interface
- Peripheral Features
  - Two 8-bit Timer/Counters with Separate Prescalers and Compare Modes
  - One 16-bit Timer/Counter with Separate Prescaler, Compare Mode, and Capture Mode
  - Real Time Counter with Separate Oscillator
  - Four PWM Channels
  - 8-channel, 10-bit ADC
    8 Single-ended Channels
    7 Differential Channels in TQFP Package Only
    2 Differential Channels with Programmable Gain at 1x, 10x, or 200x
  - Byte-oriented Two-wire Serial Interface
  - Programmable Serial USART
  - Master/Slave SPI Serial Interface
  - Programmable Watchdog Timer with Separate On-chip Oscillator
  - On-chip Analog Comparator
- Special Microcontroller Features
  - Power-on Reset and Programmable Brown-out Detection
  - Internal Calibrated RC Oscillator
  - External and Internal Interrupt Sources
  - Six Sleep Modes: Idle, ADC Noise Reduction, Power-save, Power-down, Standby and Extended Standby
- I/O and Packages
  - 32 Programmable I/O Lines
  - 40-pin PDIP, 44-lead TQFP, and 44-pad QFN/MLF
- Operating Voltages
  - 2.7V - 5.5V for ATmega16L
  - 4.5V - 5.5V for ATmega16
- Speed Grades
  - 0 - 8 MHz for ATmega16L
  - 0 - 16 MHz for ATmega16
- Power Consumption @ 1 MHz, 3V, and 25°C for ATmega16L
  - Active: 1.1 mA
  - Idle Mode: 0.35 mA
  - Power-down Mode: < 1 µA

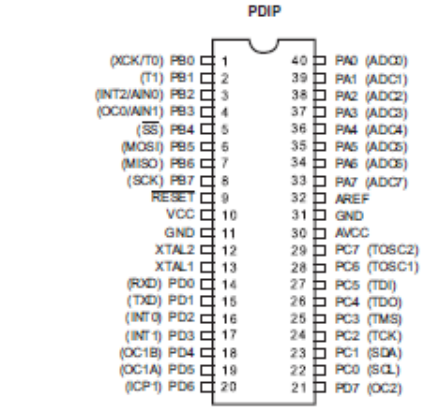
8-bit AVR® Microcontroller with 16K Bytes In-System Programmable Flash

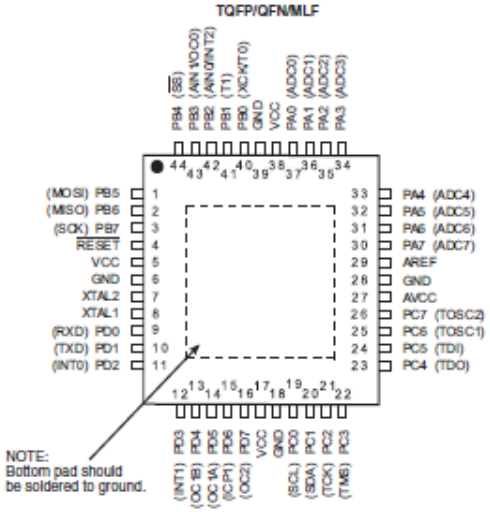
ATmega16
ATmega16L

Rev. 2466T–AVR–07/10

Pin Configurations

Figure 1. Pinout ATmega16





Disclaimer

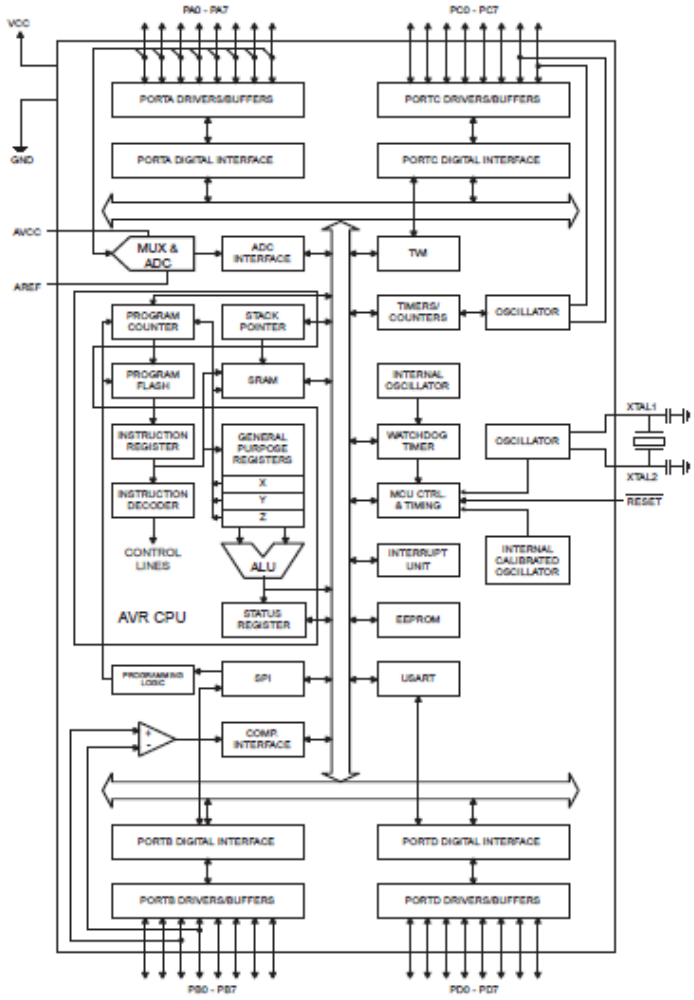
Typical values contained in this datasheet are based on simulations and characterization of other AVR microcontrollers manufactured on the same process technology. Min and Max values will be available after the device is characterized.

Overview

The ATmega16 is a low-power CMOS 8-bit microcontroller based on the AVR enhanced RISC architecture. By executing powerful instructions in a single clock cycle, the ATmega16 achieves throughputs approaching 1 MIPS per MHz allowing the system designer to optimize power consumption versus processing speed.

Block Diagram

Figure 2. Block Diagram



The AVR core combines a rich instruction set with 32 general purpose working registers. All the 32 registers are directly connected to the Arithmetic Logic Unit (ALU), allowing two independent registers to be accessed in one single instruction executed in one clock cycle. The resulting architecture is more code efficient while achieving throughputs up to ten times faster than conventional CISC microcontrollers.

The ATmega16 provides the following features: 16 Kbytes of In-System Programmable Flash Program memory with Read-While-Write capabilities, 512 bytes EEPROM, 1 Kbyte SRAM, 32 general purpose I/O lines, 32 general purpose working registers, a JTAG interface for Boundary-scan, On-chip Debugging support and programming, three flexible Timer/Counters with compare modes, Internal and External Interrupts, a serial programmable USART, a byte oriented Two-wire Serial Interface, an 8-channel, 10-bit ADC with optional differential input stage with programmable gain (TQFP package only), a programmable Watchdog Timer with Internal Oscillator, an SPI serial port, and six software selectable power saving modes. The Idle mode stops the CPU while allowing the USART, Two-wire interface, A/D Converter, SRAM, Timer/Counters, SPI port, and interrupt system to continue functioning. The Power-down mode saves the register contents but freezes the Oscillator, disabling all other chip functions until the next External Interrupt or Hardware Reset. In Power-save mode, the Asynchronous Timer continues to run, allowing the user to maintain a timer base while the rest of the device is sleeping. The ADC Noise Reduction mode stops the CPU and all I/O modules except Asynchronous Timer and ADC, to minimize switching noise during ADC conversions. In Standby mode, the crystal/resonator Oscillator is running while the rest of the device is sleeping. This allows very fast start-up combined with low-power consumption. In Extended Standby mode, both the main Oscillator and the Asynchronous Timer continue to run.

The device is manufactured using Atmel's high density nonvolatile memory technology. The On-chip ISP Flash allows the program memory to be reprogrammed in-system through an SPI serial interface, by a conventional nonvolatile memory programmer, or by an On-chip Boot program running on the AVR core. The boot program can use any interface to download the application program in the Application Flash memory. Software in the Boot Flash section will continue to run while the Application Flash section is updated, providing true Read-While-Write operation. By combining an 8-bit RISC CPU with In-System Self-Programmable Flash on a monolithic chip, the Atmel ATmega16 is a powerful microcontroller that provides a highly-flexible and cost-effective solution to many embedded control applications.

The ATmega16 AVR is supported with a full suite of program and system development tools including: C compilers, macro assemblers, program debugger/simulators, in-circuit emulators, and evaluation kits.

Pin Descriptions

VCC Digital supply voltage.

GND Ground.

Port A (PA7..PA0) Port A serves as the analog inputs to the A/D Converter.

Port A also serves as an 8-bit bi-directional I/O port, if the A/D Converter is not used. Port pins can provide internal pull-up resistors (selected for each bit). The Port A output buffers have symmetrical drive characteristics with both high sink and source capability. When pins PA0 to PA7 are used as inputs and are externally pulled low, they will source current if the internal pull-up resistors are activated. The Port A pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Port B (PB7..PB0)	Port B is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port B output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port B pins that are externally pulled low will source current if the pull-up resistors are activated. The Port B pins are tri-stated when a reset condition becomes active, even if the clock is not running. Port B also serves the functions of various special features of the ATmega16 as listed on page 58.
Port C (PC7..PC0)	Port C is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port C output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port C pins that are externally pulled low will source current if the pull-up resistors are activated. The Port C pins are tri-stated when a reset condition becomes active, even if the clock is not running. If the JTAG interface is enabled, the pull-up resistors on pins PC5(TDI), PC3(TMS) and PC2(TCK) will be activated even if a reset occurs. Port C also serves the functions of the JTAG interface and other special features of the ATmega16 as listed on page 61.
Port D (PD7..PD0)	Port D is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port D output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port D pins that are externally pulled low will source current if the pull-up resistors are activated. The Port D pins are tri-stated when a reset condition becomes active, even if the clock is not running. Port D also serves the functions of various special features of the ATmega16 as listed on page 63.
$\overline{\text{RESET}}$	Reset Input. A low level on this pin for longer than the minimum pulse length will generate a reset, even if the clock is not running. The minimum pulse length is given in Table 15 on page 38. Shorter pulses are not guaranteed to generate a reset.
XTAL1	Input to the inverting Oscillator amplifier and input to the internal clock operating circuit.
XTAL2	Output from the inverting Oscillator amplifier.
AVCC	AVCC is the supply voltage pin for Port A and the A/D Converter. It should be externally connected to V_{CC}, even if the ADC is not used. If the ADC is used, it should be connected to V_{CC} through a low-pass filter.
AREF	AREF is the analog reference pin for the A/D Converter.

Resources

A comprehensive set of development tools, application notes and datasheets are available for download on http://www.atmel.com/avr.

Data Retention

Reliability Qualification results show that the projected data retention failure rate is much less than 1 PPM over 20 years at 85°C or 100 years at 25°C.

National Semiconductor

June 1989

54154/DM54154/DM74154
4-Line to 16-Line Decoders/Demultiplexers

General Description

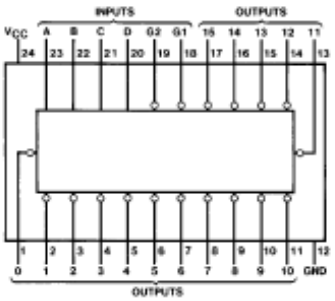
Each of these 4-line-to-16-line decoders utilizes TTL circuitry to decode four binary-coded inputs into one of sixteen mutually exclusive outputs when both the strobe inputs, G1 and G2, are low. The demultiplexing function is performed by using the 4 input lines to address the output line, passing data from one of the strobe inputs with the other strobe input low. When either strobe input is high, all outputs are high. These demultiplexers are ideally suited for implementing high-performance memory decoders. All inputs are buffered and input clamping diodes are provided to minimize transmission-line effects and thereby simplify system design.

Features

- Decodes 4 binary-coded inputs into one of 16 mutually exclusive outputs
- Performs the demultiplexing function by distributing data from one input line to any one of 16 outputs
- Input clamping diodes simplify system design
- High fan-out, low-impedance, totem-pole outputs
- Typical propagation delay
  3 levels of logic 19 ns
  Strobe 18 ns
- Typical power dissipation 170 mW
- Alternate Military/Aerospace device (54154) is available. Contact a National Semiconductor Sales Office/Distributor for specifications.

Connection Diagram

Dual-In-Line Package



TL/F/6649–1

Order Number 54154DMQB, 54154FMQB, DM54154J or DM74154N
See NS Package Number J24A, N24A or W24C

RRD-B30M115/Printed in U.S.A.

Absolute Maximum Ratings (Note)

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage	7V
Input Voltage	5.5V
Operating Free Air Temperature Range	
DM54 and 54	−55°C to +125°C
DM74	0°C to +70°C
Storage Temperature Range	−65°C to +150°C

Note: The "Absolute Maximum Ratings" are those values beyond which the safety of the device cannot be guaranteed. The device should not be operated at these limits. The parametric values defined in the "Electrical Characteristics" table are not guaranteed at the absolute maximum ratings. The "Recommended Operating Conditions" table will define the conditions for actual device operation.

Recommended Operating Conditions

Symbol	Parameter	DM54154			DM74154			Units
		Min	Nom	Max	Min	Nom	Max	
V_{CC}	Supply Voltage	4.5	5	5.5	4.75	5	5.25	V
V_{IH}	High Level Input Voltage	2			2			V
V_{IL}	Low Level Input Voltage			0.8			0.8	V
I_{OH}	High Level Output Current			−0.8			−0.8	mA
I_{OL}	Low Level Output Current			16			16	mA
T_A	Free Air Operating Temperature	−55		125	0		70	°C

Electrical Characteristics over recommended operating free air temperature range (unless otherwise noted)

Symbol	Parameter	Conditions		Min	Typ (Note 1)	Max	Units
V_I	Input Clamp Voltage	V_{CC} = Min, I_I = −12 mA				−1.5	V
V_{OH}	High Level Output Voltage	V_{CC} = Min, I_{OH} = Max, V_{IL} = Max, V_{IH} = Min		2.4	3.2		V
V_{OL}	Low Level Output Voltage	V_{CC} = Min, I_{OL} = Max, V_{IH} = Min, V_{IL} = Max			0.25	0.4	V
I_I	Input Current @ Max Input Voltage	V_{CC} = Max, V_I = 5.5V				1	mA
I_{IH}	High Level Input Current	V_{CC} = Max, V_I = 2.4V				40	µA
I_{IL}	Low Level Input Current	V_{CC} = Max, V_I = 0.4V				−1.6	mA
I_{OS}	Short Circuit Output Current	V_{CC} = Max (Note 2)	DM54	−20		−55	mA
			DM74	−18		−57	
I_{CC}	Supply Current	V_{CC} = Max (Note 3)	DM54		34	49	mA
			DM74		34	56	

Note 1: All typicals are at V_{CC} = 5V, T_A = 25°C.

Note 2: Not more than one output should be shorted at a time.

Note 3: I_{CC} is measured with all outputs open and all inputs grounded.

Switching Characteristics at V_{CC} = 5V and T_A = 25°C (See Section 1 for Test Waveforms and Output Load)

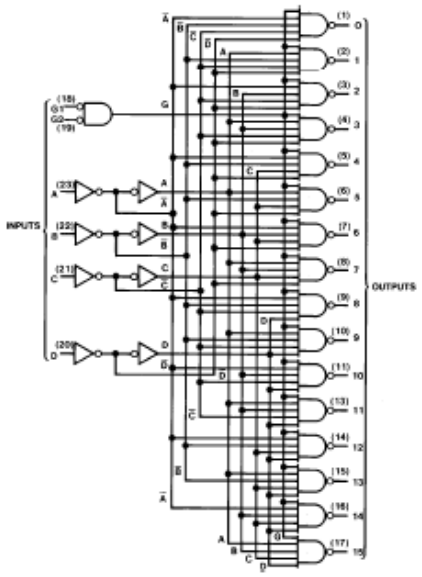
Symbol	Parameter	From (Input) To (Output)	R_L = 400Ω, C_L = 15 pF		Units
			Min	Max	
t_{PLH}	Propagation Delay Time Low to High Level Output	Data to Output		36	ns
t_{PHL}	Propagation Delay Time High to Low Level Output	Data to Output		33	ns
t_{PLH}	Propagation Delay Time Low to High Level Output	Strobe to Output		30	ns
t_{PHL}	Propagation Delay Time High to Low Level Output	Strobe to Output		27	ns

Function Table

Inputs						Outputs														
G1	G2	D	C	B	A	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
L	L	L	L	L	L	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H
L	L	L	L	L	H	H	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H
L	L	L	L	H	L	H	H	L	H	H	H	H	H	H	H	H	H	H	H	H	H
L	L	L	L	H	H	H	H	H	L	H	H	H	H	H	H	H	H	H	H	H	H
L	L	L	H	L	L	H	H	H	H	L	H	H	H	H	H	H	H	H	H	H	H
L	L	L	H	L	H	H	H	H	H	H	L	H	H	H	H	H	H	H	H	H	H
L	L	L	H	H	L	H	H	H	H	H	H	L	H	H	H	H	H	H	H	H	H
L	L	L	H	H	H	H	H	H	H	H	H	H	L	H	H	H	H	H	H	H	H
L	L	H	L	L	L	H	H	H	H	H	H	H	H	L	H	H	H	H	H	H	H
L	L	H	L	L	H	H	H	H	H	H	H	H	H	H	L	H	H	H	H	H	H
L	L	H	L	H	L	H	H	H	H	H	H	H	H	H	H	L	H	H	H	H	H
L	L	H	L	H	H	H	H	H	H	H	H	H	H	H	H	H	L	H	H	H	H
L	L	H	H	L	L	H	H	H	H	H	H	H	H	H	H	H	H	L	H	H	H
L	L	H	H	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H	L	H	H
L	L	H	H	H	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H	L	H
L	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	L
L	H	X	X	X	X	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H
H	L	X	X	X	X	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H
H	H	X	X	X	X	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H

H = High Level, L = Low Level, X = Don't Care

Logic Diagram



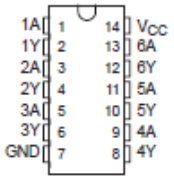
TL/F/6648-2

- **Dependable Texas Instruments Quality and Reliability**

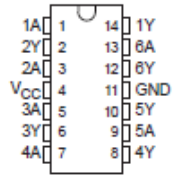
description/ordering information

These devices contain six independent inverters.

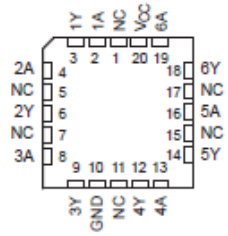
SN5404 . . . J PACKAGE
SN54LS04, SN54S04 . . . J OR W PACKAGE
SN7404, SN74S04 . . . D, N, OR NS PACKAGE
SN74LS04 . . . D, DB, N, OR NS PACKAGE
(TOP VIEW)



SN5404 . . . W PACKAGE
(TOP VIEW)



SN54LS04, SN54S04 . . . FK PACKAGE
(TOP VIEW)



NC – No internal connection

ORDERING INFORMATION

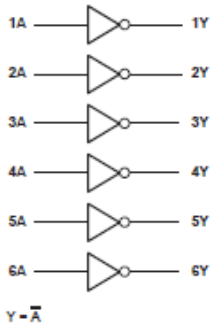
T_A	PACKAGE†		ORDERABLE PART NUMBER	TOP-SIDE MARKING
0°C to 70°C	PDIP – N	Tube	SN7404N	SN7404N
		Tube	SN74LS04N	SN74LS04N
		Tube	SN74S04N	SN74S04N
	SOIC – D	Tube	SN7404D	7404
		Tape and reel	SN7404DR	
		Tube	SN74LS04D	LS04
		Tape and reel	SN74LS04DR	
		Tube	SN74S04D	S04
		Tape and reel	SN74S04DR	
	SOP – NS	Tape and reel	SN7404NSR	SN7404
		Tape and reel	SN74LS04NSR	74LS04
		Tape and reel	SN74S04NSR	74S04
	SSOP – DB	Tape and reel	SN74LS04DBR	LS04
−55°C to 125°C	CDIP – J	Tube	SN5404J	SN5404J
		Tube	SNJ5404J	SNJ5404J
		Tube	SN54LS04J	SN54LS04J
		Tube	SN54S04J	SN54S04J
		Tube	SNJ54LS04J	SNJ54LS04J
		Tube	SNJ54S04J	SNJ54S04J
	CFP – W	Tube	SNJ5404W	SNJ5404W
		Tube	SNJ54LS04W	SNJ54LS04W
		Tube	SNJ54S04W	SNJ54S04W
	LCCC – FK	Tube	SNJ54LS04FK	SNJ54LS04FK
		Tube	SNJ54S04FK	SNJ54S04FK

† Package drawings, standard packing quantities, thermal data, symbolization, and PCB design guidelines are available at www.ti.com/sc/package.

FUNCTION TABLE
(each inverter)

INPUT A	OUTPUT Y
H	L
L	H

logic diagram (positive logic)



LAMPIRAN B
PROGRAM PADA MIKROKONTROLER ATMEGA16

```
#include <mega16.h>
#include <delay.h>

// Declare your global variables here
void huruf_M()
{
 int n,c,b,d;
 unsigned char
hMc[44]={0xF0,0xE0,0x90,0x80,0x70,0x60,0x10,0x00,0x0F,0x0E,0x0D,0x0A,0x09,0x08,0x07,0x06,
0x05,0x04,0x03,0x02,0x01,0x00,0xF0,0xE0,0xD0,0xC0,0xB0,0xA0,0x90,0x80,0x70,0x60,0x40,0x30
,0x10,0x00,0x0F,0x0E,0x09,0x08,0x07,0x06,0x01,0x00};

 for(n=1;n<129;n=n*2)
 {
      for(b=0;b<20;b++)
      {
            for(c=0;c<4;c++)
            {
                  PORTD=~n;
                  PORTA=0x07;
                  PORTC=hMc[c];
                  delay_ms(2);
            }
      }
 }

 for(n=1;n<129;n=n*2)
 {
      for(b=0;b<20;b++)
      {
            for(c=4;c<8;c++)
            {
                  PORTD=~n;
                  PORTA=0x07;
                  PORTC=hMc[c];
                  delay_ms(1);
                  PORTD=~128;
                  PORTA=0x07;
                  PORTC=hMc[c-4];
                  delay_ms(1);
            }
      }
 }

 for(n=1;n<129;n=n*2)
 {
      for(b=0;b<2;b++)
      {
            for(c=8;c<14;c++)
            {
            PORTD=~n;
```

```
            PORTA=0x0B;
            PORTC=hMc[c];
            delay_ms(6);
            }
            for(d=0;d<4;d++)
            {
                  PORTD=~128;
                  PORTA=0x07;
                  PORTC=hMc[d];
                  delay_us(3500);
                  PORTD=~128;
                  PORTA=0x07;
                  PORTC=hMc[d+4];
                  delay_us(3500);
            }
      }
}

for(n=1;n<129;n=n*2)
{
      for(b=0;b<2;b++)
      {
            for(c=14;c<22;c++)
            {
                  PORTD=~n;
                  PORTA=0x0B;
                  PORTC=hMc[c];
                  delay_ms(2);
            }
            for(d=0;d<4;d++)
            {
                  PORTD=~128;
                  PORTA=0x07;
                  PORTC=hMc[d];
                  delay_us(2500);
                  PORTD=~128;
                  PORTA=0x07;
                  PORTC=hMc[d+4];
                  delay_us(2500);
            }
            for(d=8;d<14;d++)
            {
                  PORTD=~128;
                  PORTA=0x0B;
                  PORTC=hMc[d];
                  delay_ms(4);
            }
      }
}

for(n=1;n<129;n=n*2)
```

```
{
    for(b=0;b<2;b++)
    {
        for(c=14;c<22;c++)
        {
            PORTD=~n;
            PORTA=0x0D;
            PORTB=hMc[c+8];
            delay_ms(1);
            PORTD=~128;
            PORTA=0x0B;
            PORTC=hMc[c];
            delay_ms(1);
        }
        for(d=0;d<4;d++)
        {
            PORTD=~128;
            PORTA=0x07;
            PORTC=hMc[d];
            delay_us(2500);
            PORTD=~128;
            PORTA=0x07;
            PORTC=hMc[d+4];
            delay_us(2500);
        }
        for(d=8;d<14;d++)
        {
            PORTD=~128;
            PORTA=0x0B;
            PORTC=hMc[d];
            delay_ms(4);
        }
    }
}

for(n=1;n<129;n=n*2)
{
    for(b=0;b<2;b++)
    {
        for(c=30;c<36;c++)
        {
            PORTD=~n;
            PORTA=0x0D;
            PORTB=hMc[c];
            delay_ms(2);
        }
        for(d=0;d<4;d++)
        {
            PORTD=~128;
            PORTA=0x07;
            PORTC=hMc[d];
```

```
                delay_us(2500);
                PORTD=~128;
                PORTA=0x07;
                PORTC=hMc[d+4];
                delay_us(2500);
            }
            for(d=8;d<14;d++)
            {
                PORTD=~128;
                PORTA=0x0B;
                PORTC=hMc[d];
                delay_ms(2);
            }
            for(d=14;d<22;d++)
            {
                PORTD=~128;
                PORTA=0x0D;
                PORTB=hMc[d+8];
                delay_us(1000);
                PORTD=~128;
                PORTA=0x0B;
                PORTC=hMc[d];
                delay_us(1000);
            }
        }
    }

for(n=1;n<129;n=n*2)
{
    for(b=0;b<2;b++)
    {
        for(c=0;c<4;c++)
        {
            PORTD=~n;
            PORTA=0x0E;
            PORTB=hMc[c+36];
            delay_us(1500);
            PORTD=~128;
            PORTA=0x07;
            PORTC=hMc[c];
            delay_us(1500);
            PORTD=~128;
            PORTA=0x07;
            PORTC=hMc[c+4];
            delay_us(2000);
        }
        for(d=14;d<22;d++)
        {
            PORTD=~128;
            PORTA=0x0D;
            PORTB=hMc[d+8];
```

```
                delay_ms(1);
                PORTD=~128;
                PORTA=0x0B;
                PORTC=hMc[d];
                delay_ms(1);
            }
            for(d=8;d<14;d++)
            {
                PORTD=~128;
                PORTA=0x0B;
                PORTC=hMc[d];
                delay_ms(2);
            }
            for(d=30;d<36;d++)
            {
                PORTD=~128;
                PORTA=0x0D;
                PORTB=hMc[d];
                delay_s(2);
            }
        }
}

 for(b=0;b<900;b++)
 {
 for(c=0;c<4;c++)
 {
          PORTD=~128;
          PORTA=0x0E;
          PORTB=hMc[c+40];
          delay_ms(1);
           PORTD=~128;
          PORTA=0x0E;
          PORTB=hMc[c+36];
          delay_ms(1);
          PORTD=~128;
          PORTA=0x07;
          PORTC=hMc[c];
          delay_ms(1);
          PORTD=~128;
          PORTA=0x07;
          PORTC=hMc[c+4];
           delay_ms(1);
 }
 for(d=8;d<14;d++)
  {
          PORTD=~128;
          PORTA=0x0B;
          PORTC=hMc[d];
          delay_ms(2);
  }
```

```
 for(d=14;d<22;d++)
 {
          PORTD=~128;
          PORTA=0x0D;
          PORTB=hMc[d+8];
          delay_ms(1);
          PORTD=~128;
          PORTA=0x0B;
          PORTC=hMc[d];
          delay_ms(1);
 }
 for(d=30;d<36;d++)
 {
          PORTD=~128;
          PORTA=0x0D;
          PORTB=hMc[d];
          delay_ms(2);
 }
 }
}

 void huruf_A()
 {
  int i,n,b,a,k,l,j;
  unsigned char hA[4]={0x07,0x06,0x01,0x00};
  unsigned char
atas[24]={0x08,0x19,0x2A,0x3B,0x4C,0x5D,0x6E,0x7F,0x00,0x01,0x06,0x07,0x80,0x90,0xA0,0xB0
,0xC0,0xD0,0xE0,0xF0,0x08,0x19,0x6E,0x7F};
  unsigned char
atas2[24]={0x00,0x10,0x20,0x30,0x40,0x50,0x60,0x70,0x80,0x90,0xA0,0xB0,0xC0,0xD0,0xE0,0xF
0,0x00,0x01,0x06,0x07,0x08,0x09,0x0E,0x0F};
  for(n=3;n<193;n=n*4)
  {
      for(b=0;b<2;b++)
      {
            for(i=0;i<8;i++)
            {
                  PORTD=~n;
                  PORTA=0x0E;
                  PORTB=i;
                  delay_ms(8);
            }
      }
  }

  for(n=3;n<49;n=n*4)
  {
      for(b=0;b<2;b++)
      {
            for(i=0;i<4;i++)
            {
```

```
                PORTD=~n;
                PORTA=0x0E;
                PORTB=hA[i];
                delay_ms(8);
            }
            for(a=0;a<8;a++)
            {
                PORTD=~192;
                PORTA=0x0E;
                PORTB=a;
                delay_ms(4);
            }
        }
    }

    for(n=3;n<14;n=n*4)
    {
        for(b=0;b<2;b++)
        {
            for(i=0;i<8;i++)
            {
                PORTD=~n;
                PORTA=0x0E;
                PORTB=i;
                delay_ms(2);
                PORTD=~192;
                PORTA=0x0E;
                PORTB=i;
                delay_ms(2);
            }
            for(i=0;i<4;i++)
            {
                PORTD=~48;
                PORTA=0x0E;
                PORTB=hA[i];
                delay_ms(8);
            }
        }
    }

    for(b=0;b<2;b++)
    {
        for(i=0;i<4;i++)
        {
            PORTD=~3;
            PORTA=0x0E;
            PORTB=hA[i];
            delay_ms(8);
        }
        for(k=0;k<8;k++)
        {
```

```
            PORTD=~12;
            PORTA=0x0E;
            PORTB=k;
            delay_ms(1);
        }
 for(a=0;a<8;a++)
{
          PORTD=~192;
          PORTA=0x0E;
          PORTB=a;
          delay_ms(1);
}
 for(l=0;l<4;l++)
{
           PORTD=~48;
          PORTA=0x0E;
           PORTB=hA[l];
          delay_ms(4);
 }
 }

 for(i=0;i<2;i++)
 {
     for(j=0;j<8;j++)
     {
          PORTD=~128;
          PORTA=0x0C;
          PORTB=atas[j];
          delay_ms(1);
     }
     for(b=8;b<12;b++)
     {
          PORTD=0b00111111;
          PORTA=0x0E;
          PORTB=atas[b];
          delay_ms(4);
     }
     for(b=0;b<8;b++)
     {
          PORTD=0b11001111;
          PORTA=0x0E;
          PORTB=b;
          delay_ms(1);
     }
     for(b=8;b<12;b++)
     {
          PORTD=0b11110011;
          PORTA=0x0E;
          PORTB=atas[b];
          delay_ms(4);
```

```
    }
}
for(i=0;i<2;i++)
{
    for(j=12;j<20;j++)
    {
        PORTD=~128;
        PORTA=0x09;
        PORTB=atas[j];
        PORTC=j-12;
        delay_ms(1);
    }
    for(j=20;j<24;j++)
    {
        PORTD=~128;
        PORTA=0x0C;
        PORTB=atas[j];
        delay_ms(2);
    }
    for(j=0;j<8;j++)
    {
        PORTD=0b00111111;
        PORTA=0x0E;
        PORTB=j;
        delay_ms(1);
    }
    for(j=8;j<12;j++)
    {
        PORTD=0b11001111;
        PORTA=0x0E;
        PORTB=atas[j];
        delay_ms(2);
    }
}

for(i=0;i<30;i++)
{
    for(j=0;j<16;j++)
    {
        PORTD=~128;
        PORTA=0x05;
        PORTB=atas2[j];
        PORTC=atas2[j];
        delay_ms(2);
    }
    for(j=16;j<24;j++)
    {
        PORTD=~128;
        PORTA=0x0A;
        PORTB=atas2[j];
        PORTC=atas2[j];
```

```
            delay_ms(4);
        }
 }
}

void huruf_R()
{
int i,j;

for(i=0;i<2;i++)
{
      for(j=0;j<5;j++)
      {
            PORTD=0x01;
            PORTA=0x07;
            PORTC=0x70;
            delay_ms(12);
            PORTD=0x6F;
            PORTA=0x0B;
            PORTC=0x0F;
            delay_ms(12);
            PORTD=0x67;
            PORTA=0x0B;
            PORTC=0x07;
            delay_ms(12);
            PORTD=0x6B;
            PORTA=0x0D;
            PORTB=0xF0;
            delay_ms(12);
            PORTD=0x9D;
            PORTA=0x0D;
            PORTB=0x70;
            delay_ms(12);
      }
}

for(i=0;i<20;i++)
{
      for(j=0;j<5;j++)
      {
            PORTD=0x01;
            PORTA=0x07;
            PORTC=0x50;
            delay_ms(12);
            PORTD=0x6F;
            PORTA=0x0B;
            PORTC=0x0D;
            delay_ms(12);
            PORTD=0x67;
            PORTA=0x0B;
            PORTC=0x05;
```

```
            delay_ms(12);
            PORTD=0x6B;
            PORTA=0x0D;
            PORTB=0xD0;
            delay_ms(12);
            PORTD=0x9D;
            PORTA=0x0D;
            PORTB=0x50;
            delay_ms(12);
      }
}

for(i=0;i<20;i++)
{
      for(j=0;j<5;j++)
      {
            PORTD=0x01;
            PORTA=0x07;
            PORTC=0x30;
            delay_ms(12);
            PORTD=0x6F;
            PORTA=0x0B;
            PORTC=0x0B;
            delay_ms(12);
            PORTD=0x67;
            PORTA=0x0B;
            PORTC=0x03;
            delay_ms(12);
            PORTD=0x6B;
            PORTA=0x0D;
            PORTB=0xB0;
            delay_ms(12);
            PORTD=0x9D;
            PORTA=0x0D;
            PORTB=0x30;
            delay_ms(12);
      }
}

for(i=0;i<20;i++)
{
      for(j=0;j<5;j++)
      {
            PORTD=0x01;
            PORTA=0x07;
            PORTC=0x00;
            delay_ms(12);
            PORTD=0x6F;
            PORTA=0x0B;
            PORTC=0x08;
            delay_ms(12);
```

```
            PORTD=0x67;
            PORTA=0x0B;
            PORTC=0x00;
            delay_ms(12);
            PORTD=0x6B;
            PORTA=0x0D;
            PORTB=0x80;
            delay_ms(12);
            PORTD=0x9D;
            PORTA=0x0D;
            PORTB=0x00;
            delay_ms(12);
        }
    }
    }

    void huruf_N()
    {
    int i,j,k,l;
    unsigned char hnb[9]={0x30,0x50,0xB0,0xC0,0xD0,0x20,0x60,0xA0,0xE0};
    unsigned char hnc[9]={0x02,0x03,0x04,0x05,0x06,0x0A,0x0B,0x0D,0x0E};
    unsigned char
   hNb[13]={0x00,0x11,0x66,0x77,0x88,0x99,0x2A,0xEE,0xFF,0x30,0xA0,0xB0,0xC0};
    unsigned char hNc[13]={0x00,0x11,0x66,0x77,0x88,0x99,0x0C,0x5D,0xEE,0xFF,0x55,0x03,0x14};
    PORTD=~16;
    PORTA=0x0D;
    PORTB=0xC0;
    delay_ms(500);

    for(j=0;j<150;j++)
    {
        for(i=0;i<5;i++)
        {
            PORTD=~16;
            PORTA=0x09;
            PORTB=hnb[i];
            PORTC=0x03;
            delay_ms(6);
            PORTC=0x05;
            delay_ms(6);
        }
    }

    for(j=0;j<10;j++)
    {
        for(i=0;i<9;i++)
        {
            PORTD=~16;
            PORTA=0x09;
            PORTB=hnb[i];
            PORTC=hnc[i];
```

```
            delay_ms(7);
        }
}

for(j=0;j<10;j++)
{
    for(i=0;i<13;i++)
    {
            PORTD=~16;
            PORTA=0x00;
            PORTB=hNb[i];
            PORTC=hNc[i];
            delay_ms(5);
    }
}


    for(k=8;k<129;k=k*2)
    {
    for(l=0;l<2;l++)
    {
    for(i=0;i<13;i++)
    {
            PORTD=~(k*2);
            PORTA=0x00;
            PORTB=hNb[i];
            PORTC=hNc[i];
            delay_ms(5);
    }
    }

    }

}

void huruf_T()
{
 int i,j;
 unsigned char
hT[22]={0x19,0x2A,0x3B,0x4C,0x5D,0x6E,0x90,0xA0,0xB0,0xC0,0xD0,0xE0,0x3B,0x4C,0x99,0x
AA,0xBB,0xCC,0xDD,0xEE,0xB3,0xC4};

 for(j=0;j<3;j++)
 {
 for(i=1;i<7;i++)
 {
    PORTD=0b11111100;
    PORTA=0x0E;
    PORTB=i;
    delay_ms(10);
 }
```

```
}
for(j=0;j<3;j++)
{
for(i=1;i<7;i++)
{
    PORTD=0b11110001;
    PORTA=0x0E;
    PORTB=i;
    delay_ms(5);
}
for(i=3;i<5;i++)
{
    PORTD=0b11111110;
    PORTA=0x0E;
    PORTB=i;
    delay_ms(15);
}
}

for(j=0;j<3;j++)
{
    for(i=1;i<7;i++)
    {
        PORTD=0b00011111;
        PORTA=0x0E;
        PORTB=i;
        delay_ms(5);
    }
    for(i=3;i<5;i++)
    {
        PORTD=0b11100000;
        PORTA=0x0E;
        PORTB=i;
        delay_ms(15);
    }
}

for(j=0;j<3;j++)
{
    for(i=0;i<6;i++)
    {
        PORTD=~128;
        PORTA=0x0C;
        PORTB=hT[i];
        delay_us(2500);
        PORTB=i;
        delay_us(2500);
     }
    for(i=3;i<5;i++)
    {
        PORTD=0b10000011;
```

```
            PORTA=0x0E;
            PORTB=i;
            delay_ms(15);
        }
    }
    for(j=0;j<3;j++)
    {
        for(i=0;i<6;i++)
        {
            PORTD=~128;
            PORTA=0x0C;
            PORTB=hT[i];
            delay_us(2500);
            PORTA=0x0D;
            PORTB=hT[i+6];
            delay_us(2500);
        }
        for(i=3;i<5;i++)
        {
            PORTD=0b00000111;
            PORTA=0x0E;
            PORTB=i;
            delay_ms(1);
        }
    }
    for(j=0;j<3;j++)
    {
        for(i=6;i<12;i++)
        {
            PORTD=~128;
            PORTA=0x09;
            PORTB=hT[i];
            PORTC=hT[i];
            delay_us(2500);
            PORTA=0x0B;
            PORTC=i+3;
            delay_us(2500);
        }
        for(i=12;i<14;i++)
        {
            PORTD=~128;
            PORTA=0x0C;
            PORTB=hT[i];
            delay_us(7500);
            PORTD=0b00011111;
            PORTA=0x0E;
            PORTB=i-9;
            delay_us(7500);
        }
    }
    for(j=0;j<3;j++)
```

```
{
      for(i=14;i<20;i++)
      {
            PORTD=~128;
            PORTA=0x03;
            PORTC=hT[i];
            delay_us(2500);
            PORTA=0x07;
            PORTC=hT[i-14];
            delay_us(2500);
      }
      for(i=20;i<22;i++)
      {
            PORTA=0x0C;
            PORTB=hT[i];
            delay_us(7500);
            PORTA=0x08;
            PORTC=i-17;
            PORTB=hT[i-8];
            delay_us(7500);
      }
}
}

 void huruf_H()
 {
 int i,j;
 unsigned char
hH[32]={0xF0,0xE0,0xD0,0xC0,0xB0,0xA0,0x90,0x80,0x70,0x60,0x50,0x40,0x30,0x20,0x10,0x00,
0x0F,0x0E,0x0D,0x0C,0x0B,0x0A,0x09,0x08,0x07,0x06,0x05,0x04,0x03,0x02,0x01,0x00};
 unsigned char PD[8]={0x00,0x00,0xE7,0xE7,0xE7,0xE7,0x00,0x00};
 //unsigned char PA[8]={0x07,0x07,0x0B,0x0B,0x0D,0x0D,0x0E,0x0E};

for(i=0;i<3;i++)
{
      for(j=0;j<8;j++)
      {
            PORTD=PD[j];
            PORTA=0x07;
            PORTC=hH[j];
            delay_ms(8);
      }
}

for(i=0;i<3;i++)
{
      for(j=8;j<16;j++)
      {
            PORTD=PD[j-8];
            PORTA=0x07;
            PORTC=hH[j];
```

```
            delay_ms(8);
        }
}

for(i=0;i<3;i++)
{
        for(j=16;j<24;j++)
        {
            PORTD=PD[j-16];
            PORTA=0x0B;
            PORTC=hH[j];
            delay_ms(8);
        }
}

for(i=0;i<3;i++)
{
        for(j=24;j<32;j++)
        {
            PORTD=PD[j-24];
            PORTA=0x0B;
            PORTC=hH[j];
            delay_ms(8);
        }
}

for(i=0;i<3;i++)
{
        for(j=0;j<8;j++)
        {
            PORTD=PD[j];
            PORTA=0x0D;
            PORTB=hH[j];
            delay_ms(8);
        }
}

for(i=0;i<3;i++)
{
        for(j=8;j<16;j++)
        {
            PORTD=PD[j-8];
            PORTA=0x0D;
            PORTB=hH[j];
            delay_ms(8);
        }
}

for(i=0;i<3;i++)
{
        for(j=16;j<24;j++)
```

```
        {
            PORTD=PD[j-16];
            PORTA=0x0E;
            PORTB=hH[j];
            delay_ms(8);
        }
}

for(i=0;i<5;i++)
{
        for(j=24;j<32;j++)
        {
            PORTD=PD[j-24];
            PORTA=0x0E;
            PORTB=hH[j];
            delay_ms(8);
        }
}

 }

 void kembang_api()
 {
 unsigned char Kc1[12]={0x67,0x62,0xF8,0xC6,0xE5,0x63,0x04A,0xA0,0x41,0x12,0x33,0x7D};
 unsigned char
Kc2[13]={0x75,0x26,0xAA,0x9B,0xF8,0xDB,0x6C,0xC0,0x72,0x8C,0x2A,0x63,0x44};
 int i,v,c;

 for(i=1;i<129;i=i*2)
 {
        PORTD=~i;
        PORTA=0x0B;
        PORTC=0x05;
        delay_ms(200);
 }
 for(i=128;i>=32;i=i/2)
 {
        for(c=0;c<7;c++)
        {
            for(v=8;v<12;v++)
            {
                PORTD=~i;
                PORTA=0x0A;
                PORTC=Kc1[v];
                PORTB=Kc1[v];
                delay_ms(1);
            }
            for(v=0;v<3;v++)
            {
                PORTD=~(i/2);
                PORTA=0x05;
```

```
                PORTC=Kc1[v];
                PORTB=Kc1[v];
                delay_ms(1);
            }
        }

}
for(i=128;i>0;i=i/2)
{
    for(c=0;c<7;c++)
    {
        for(v=8;v<12;v++)
        {
            PORTD=~(i/8);
            PORTA=0x0A;
            PORTC=Kc1[v];
            PORTB=Kc1[v];
            delay_ms(1);
        }
        for(v=0;v<3;v++)
        {
            PORTA=0x05;
            PORTC=Kc1[v];
            PORTB=Kc1[v];
            delay_ms(1);
        }
        for(v=3;v<7;v++)
        {
            PORTD=~i;
            PORTA=0x05;
            PORTB=Kc1[v];
            PORTC=Kc2[v];
            delay_ms(1);
        }

    }
}

 for(i=1;i<129;i=i*2)
{
 PORTD=~i;
 PORTA=0x0D;
 PORTB=0xA0;
 delay_ms(300);
}
for(i=128;i>=16;i=i/2)
{
    for(c=0;c<7;c++)
    {
        for(v=7;v<12;v++)
        {
```

```
                PORTD=~i;
                PORTA=0x0A;
                PORTC=Kc2[v];
                PORTB=Kc2[v];
                delay_ms(1);
            }
            for(v=2;v<4;v++)
            {
                PORTA=0x05;
                PORTC=Kc2[v];
                PORTB=Kc2[v];
                delay_ms(1);
            }

        }
    }

    for(i=128;i>60;i=i/2)
    {
        for(c=0;c<7;c++)
        {
            for(v=7;v<12;v++)
            {
                PORTD=~(i/16);
                PORTA=0x0A;
                PORTC=Kc2[v];
                PORTB=Kc2[v];
                delay_ms(1);
            }
            for(v=2;v<4;v++)
            {
                PORTA=0x05;
                PORTC=Kc2[v];
                PORTB=Kc2[v];
                delay_ms(1);
            }
            for(v=5;v<8;v++)
            {
                PORTD=~i;
                PORTA=0x0A;
                PORTC=Kc1[v];
                PORTB=Kc1[v];
                delay_ms(1);
            }
        }
    }

    for(i=128;i>0;i=i/2)
    {
        for(c=0;c<7;c++)
        {
```

```
            for(v=7;v<12;v++)
            {
                  PORTD=~(i/32);
                  PORTA=0x0A;
                  PORTC=Kc2[v];
                  PORTB=Kc2[v];
                  delay_ms(1);
            }
            for(v=2;v<4;v++)
            {
                  PORTA=0x05;
                  PORTC=Kc2[v];
                  PORTB=Kc2[v];
                  delay_ms(1);
            }
            for(v=5;v<8;v++)
            {
                  PORTD=~(i/4);
                  PORTA=0x0A;
                  PORTC=Kc1[v];
                  PORTB=Kc1[v];
                  delay_ms(1);
            }
            for(v=4;v<7;v++)
            {
                  PORTD=~i;
                  PORTA=0x05;
                  PORTC=Kc1[v];
                  PORTB=Kc1[v];
                  delay_ms(1);
            }
      }
}

for(i=1;i<129;i=i*2)
{
 PORTD=~i;
 PORTA=0x07;
 PORTC=0x40;
 delay_ms(300);
}
for(i=128;i>=1;i=i/2)
{
      for(c=0;c<7;c++)
      {
            for(v=2;v<5;v++)
            {
                  PORTD=~i;
                  PORTA=0x0A;
                  PORTC=Kc1[v];
                  PORTB=Kc2[v];
```

```
                delay_ms(1);
            }
            for(v=6;v<11;v++)
            {
                PORTD=~(i/2);
                PORTA=0x05;
                PORTC=Kc2[v];
                PORTB=Kc1[v];
                delay_ms(1);
            }

    }
}
}

void hujan()
{
int i,j,k,l;

for(l=0;l<2;l++)
{
for(i=128;i>30;i=i/2)
{
    for(j=0;j<70;j++)
    {
        PORTD=~i;
        PORTA=0x00;
        PORTB=0xD7;
        PORTC=0xA3;
        delay_ms(1);
        PORTD=~i;
        PORTA=0x0A;
        PORTB=0x09;
        PORTC=0x0E;
        delay_ms(1);
    }
}

for(i=128;i>16;i=i/2)
{
    for(k=0;k<11;k++)
    {
    for(j=0;j<5;j++)
    {
        PORTD=~(i/8);
        PORTA=0x00;
        PORTB=0xD7;
        PORTC=0xA3;
        delay_us(500);
        PORTD=~(i/8);
        PORTA=0x0A;
```

```
            PORTB=0x09;
            PORTC=0x0E;
            delay_us(500);
        }
        for(j=0;j<5;j++)
        {
            PORTD=~i;
            PORTA=0x00;
            PORTB=0x04;
            PORTC=0x45;
            delay_ms(1);
            PORTD=~i;
            PORTA=0x00;
            PORTB=0x3E;
            PORTC=0xF8;
            delay_ms(1);
        }
        }
}

for(i=128;i>60;i=i/2);
{
    for(k=0;k<8;k++)
    {
    for(j=0;j<5;j++)
    {
            PORTD=~(i/16);
            PORTA=0x00;
            PORTB=0x04;
            PORTC=0x45;
            delay_ms(1);
            PORTD=~(i/16);
            PORTA=0x00;
            PORTB=0x3E;
            PORTC=0xF8;
            delay_ms(1);
    }
    for(j=0;j<5;j++)
    {
            PORTD=~i;
            PORTA=0x00;
            PORTB=0x6C;
            PORTC=0x70;
            delay_ms(1);
            PORTD=~i;
            PORTA=0x00;
            PORTB=0x50;
            PORTC=0xC2;
            delay_ms(1);
    }
    }
```

```
}

for(i=128;i>0;i=i/2)
{
    for(k=0;k<8;k++)
    {
    for(j=0;j<5;j++)
    {
        PORTD=~(i/4);
        PORTA=0x00;
        PORTB=0x6C;
        PORTC=0x70;
        delay_ms(1);
        PORTD=~(i/4);
        PORTA=0x00;
        PORTB=0x50;
        PORTC=0xC2;
        delay_ms(1);
    }
    for(j=0;j<5;j++)
    {
        PORTD=~i;
        PORTA=0x00;
        PORTB=0xC4;
        PORTC=0x9C;
        delay_ms(1);
        PORTD=~i;
        PORTA=0x00;
        PORTB=0x46;
        PORTC=0xA5;
        delay_ms(1);
    }
    }
}
}
}

void main(void)
{
// Declare your local variables here

// Input/Output Ports initialization
// Port A initialization
// Func7=Out Func6=Out Func5=Out Func4=Out Func3=Out Func2=Out Func1=Out Func0=Out
// State7=0 State6=0 State5=0 State4=0 State3=0 State2=0 State1=0 State0=0
PORTA=0x00;
DDRA=0xFF;

// Port B initialization
// Func7=Out Func6=Out Func5=Out Func4=Out Func3=Out Func2=Out Func1=Out Func0=Out
// State7=0 State6=0 State5=0 State4=0 State3=0 State2=0 State1=0 State0=0
```

```
PORTB=0x00;
DDRB=0xFF;

// Port C initialization
// Func7=Out Func6=Out Func5=Out Func4=Out Func3=Out Func2=Out Func1=Out Func0=Out
// State7=0 State6=0 State5=0 State4=0 State3=0 State2=0 State1=0 State0=0
PORTC=0x00;
DDRC=0xFF;

// Port D initialization
// Func7=Out Func6=Out Func5=Out Func4=Out Func3=Out Func2=Out Func1=Out Func0=Out
// State7=0 State6=0 State5=0 State4=0 State3=0 State2=0 State1=0 State0=0
PORTD=0x00;
DDRD=0xFF;

// Timer/Counter 0 initialization
// Clock source: System Clock
// Clock value: Timer 0 Stopped
// Mode: Normal top=FFh
// OC0 output: Disconnected
TCCR0=0x00;
TCNT0=0x00;
OCR0=0x00;

// Timer/Counter 1 initialization
// Clock source: System Clock
// Clock value: Timer 1 Stopped
// Mode: Normal top=FFFFh
// OC1A output: Discon.
// OC1B output: Discon.
// Noise Canceler: Off
// Input Capture on Falling Edge
// Timer 1 Overflow Interrupt: Off
// Input Capture Interrupt: Off
// Compare A Match Interrupt: Off
// Compare B Match Interrupt: Off
TCCR1A=0x00;
TCCR1B=0x00;
TCNT1H=0x00;
TCNT1L=0x00;
ICR1H=0x00;
ICR1L=0x00;
OCR1AH=0x00;
OCR1AL=0x00;
OCR1BH=0x00;
OCR1BL=0x00;

// Timer/Counter 2 initialization
// Clock source: System Clock
// Clock value: Timer 2 Stopped
// Mode: Normal top=FFh
```

```
// OC2 output: Disconnected
ASSR=0x00;
TCCR2=0x00;
TCNT2=0x00;
OCR2=0x00;

// External Interrupt(s) initialization
// INT0: Off
// INT1: Off
// INT2: Off
MCUCR=0x00;
MCUCSR=0x00;

// Timer(s)/Counter(s) Interrupt(s) initialization
TIMSK=0x00;

// Analog Comparator initialization
// Analog Comparator: Off
// Analog Comparator Input Capture by Timer/Counter 1: Off
ACSR=0x80;
SFIOR=0x00;

while (1)
     {
     // Place your code here
       kembang_api();
       PORTD=255;
       delay_ms(2000);
       hujan();
       PORTD=255;
       delay_ms(1000);
       huruf_M();
       huruf_A();
       huruf_R();
       huruf_A();
       huruf_N();
       huruf_A();
       huruf_T();
       huruf_H();
       huruf_A();
     };
}
```