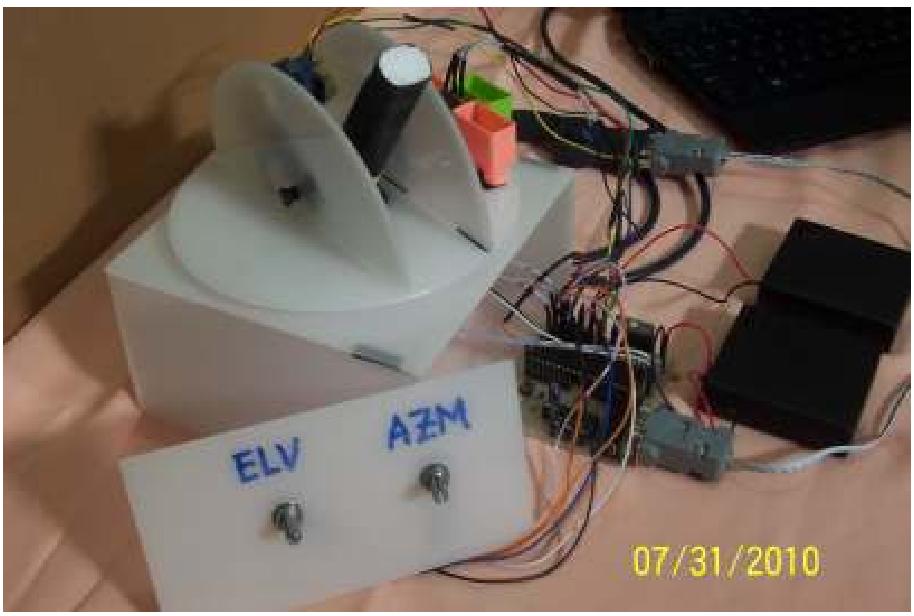
LAMPIRAN A

FOTO ALAT

PROTOTYPE MERIAM



LAMPIRAN B

PROGRAM AVR STUDIO

```
/*
;*************************************************************
;* File            : TaMikro.c
;* Title           : Pemrograman uC (ADC, UART, Motor PWM)
;* Version         : 1.00
;* Date            : 1 Jun 2010
;* Target          : ATMega8535
;* Author          : (c) 2010 - Septi
;* Program         : Sistem Kendali Meriam
;*************************************************************
;* DESCRIPTION
;*  Aplikasi :      - Komunikasi dengan PC via Rs232 in 9600 bps
;*                  - Baca Data Input ADC
;*                  - Aktifkan PWM Motor to L298
;**************************************************************************
*/

#include <avr/io.h>
#include <avr/interrupt.h>
#include <compat/deprecated.h>

#define _NOP() do { __asm__ __volatile__ ("nop"); } while (0)

// Prescaler :
#define kPrescaler 0x02                         // Prescaler = 8
#define kTime100        230                     // *100 = 60Hz

// Baud Rate definition (U2X = 0) :
#define kUbrr 0x47                              // 0x47 = 71 = 9600 bps

// ---------------------------------------------------------------------------------------------------------------
#define ADC_REF 0x00                            // ADC ARef, Right Data
#define kHH 0x10                                // Header
#define kTT 0x03                                // Tail
#define kMaxTx 16                               // Data Tx : uC to PC
#define kMaxRx 8                                // Data Rx : PC to uC
#define kMaxAdc 4                               // Data ADC

#define kDeadZone 3                             // Dead Zone untuk Transisi
unsigned char kPwmAz=50;                        // Default Pwm Az = 50%
unsigned char kPwmEl=50;                        // Default Pwm El = 50%
unsigned char kAkurasiAz=15;                    // Akurasi Steady State Az
unsigned char kAkurasiEl=15;                    // Akurasi Steady State El

unsigned char cDataTx = 'A';
unsigned char cDataRx = 'A';

unsigned char iAdcNo = 0;
unsigned char cAdcH = 0;
unsigned char cAdcL = 0;

unsigned char iCountRx = 0;
unsigned char iCountTx = 0;
unsigned char iTxNo;
unsigned char iRxNo = 0;
unsigned char cByteTx[kMaxTx];
unsigned char cByteRx[kMaxRx];
```

```
unsigned char bPcOk = 0x00;
unsigned char cPortC = 0x00;
unsigned char cPortB = 0x00;

unsigned int iTimePwm = 0;
unsigned int iTimeAz = 512;
unsigned int iTimeEl = 512;

unsigned int iSensor[kMaxAdc];                    // Sensor ADC di Port A
unsigned int iPcInAz;                             // Data dari PC - Azimuth
unsigned int iPcInEl;                             // Data dari PC - Elevasi
unsigned int iDeltaAz = 0;
unsigned int iDeltaEl = 0;

unsigned char cTimePwm = 0;
unsigned char cTimeAz = 0;
unsigned char cTimeEl = 0;
unsigned char cTimeTx = 0;
unsigned char cStatusAz = 0x00;
unsigned char cStatusEl = 0x00;


//===========================================================================
// ROUTINES
//===========================================================================

// Routine : Timer 1 Overflow Vector INTERRUPT
INTERRUPT (TIMER1_OVF_vect)
{
        TCNT1 = -kTime100;
        cTimePwm++;
        if (cTimePwm >= 100)
        {
                cTimePwm = 0x00;
                cTimeTx++;
        }

        if (cTimeAz>cTimePwm)
        { cPortC |= 0x04; }                       // C2 Pwm
        else
        { cPortC &= 0xFB; }                       // C2 Low

        if (cTimeEl>cTimePwm)
        { cPortC |= 0x80; }                       // C7 Pwm
        else
        { cPortC &= 0x7F; }                       // C7 Low

        PORTC = cPortC;
}


// Fungsi : Inisialisasi Hardware
//---------------------------------------------------------------------------
void PInitHW(void)
{
```

```
		// Port : A, B, C, D
		DDRA = 0x00;				// All PortA sebagai INPUT
		PORTA = 0x00;				// Port A : Konfigurasi High Impedance
		_NOP();

		DDRB = 0x00;				// All Port B sebagai INPUT
		PORTB = 0xFF;				// Port B : Konfigurasi Pull up
		_NOP();

		DDRC = 0xFF;				// All Port C sebagai OUTPUT
		PORTC = 0x00;				// Port C : Konfigurasi High Impedance
		_NOP();

//		DDRD = (DDRD & 0x0F) | 0xC0;	// All Port D sebagai INPUT 5-6, OUTPUT 7-8
//		PORTD = PORTD & 0x0F;			// Port D : Konfigurasi High Impedance 5-8
//		_NOP();

		TCCR1A = 0x00;
		TCCR1B = kPrescaler;			// See Prescaler definition
		TCNT1H = 0x00;
		TCNT1L = 0x00;
		ICR1H = 0x00;
		ICR1L = 0x00;
		OCR1AH = 0x00;
		OCR1AL = 0x00;
		OCR1BH = 0x00;
		OCR1BL = 0x00;

		ASSR = 0x00;
		TCCR2 = 0x07;
		TCNT2 = 0x00;
		OCR2 = 0x00;

// External Interrupt(s) initialization
		MCUCR = 0x00;
		MCUCSR = 0x00;
		TIMSK = 0x04;				// Timer(s)/Counter(s) Overflow Interrupt(s)

// Ready : ADC (Analog to Digital Converter)
		ACSR = 0x80;				// Analog Comparator initialization
		ADMUX = ADC_REF;
		ADCSRA = 0x83;				// ADC Enabled, DivFactor = 64
		SFIOR = 0xEF;

// USART Initialization : 4800,n,8,1
		UCSRA = 0x00;			// U2X=0 : Baud rate divider (ASync Only)
		UCSRB = 0x18;			// Tx + Rx : ON
		UCSRC = 0x86;			// Write, Asynchronous, No Parity, Stop bit = 1, 8bit
		UBRRH = 0x00;			// Ready for Writing UBRR
		UBRRL = kUbrr;			// See Constant Definition
}
void PInitData(void)
{
		iTxNo = 0;
		iAdcNo = 0;
		cByteTx[0] = 0x10;
		cByteTx[1] = 'A';
```

```
	cByteTx[2] = 1;
	cByteTx[3] = 'C';
	cByteTx[4] = 1;
	cByteTx[5] = 'E';
	cByteTx[6] = 'F';
	cByteTx[7] = 0x3;

	iSensor[0] = 512;
	iSensor[1] = 512;
	iSensor[2] = 512;
	iSensor[3] = 512;

	iPcInAz = 512;
	iPcInEl = 512;
}

// Fungsi : Kirim Data ke PC
//-----------------------------------------------------------------------------------------------------------------
void PDataTx(void)
{
	if (UCSRA & (1<<UDRE))                       // Menunggu buffer Tx kosong
	{
		cDataTx = cByteTx[iTxNo];            // Serial Tx : char by char
		UDR = cDataTx;                       // Isi Buffer TX
		iTxNo++;
	}
}
// Fungsi : Terima Data dari PC
//-----------------------------------------------------------------------------------------------------------------
void PDataRx(void)
{
	cDataRx = UDR;                               // Baca Buffer RX
	cByteRx[iRxNo] = cDataRx;                    // Serial Rx : char by char
	iRxNo++;                                     // Query : Ascending
	if (iRxNo == 0x01)                           // Data Rx : Check Header
	{
		if (cDataRx != 0x10) iRxNo--;
	}
	else if (iRxNo >= 8)                         // Data Rx : Check Tail
	{
	    if (cDataRx == 0x03)                     // Tail Accepted
	    {
		if (cByteRx[1] == 0x05)              // Ack : Data Konfigurasi
		{
		    kAkurasiAz = cByteRx[2];         // Akurasi Az
		    kAkurasiEl = cByteRx[3];         // Akurasi El
		    kPwmAz = cByteRx[4];             // Pwm Speed Az
		    kPwmEl = cByteRx[5];             // Pwm Speed El
		}
	    else                                     // Ack : Data Reference
	    {
		iPcInAz = (cByteRx[2] << 8) | cByteRx[3];
		iPcInEl = (cByteRx[4] << 8) | cByteRx[5];
	    }
    }
	iRxNo = 0x00;                                // Data Rx : Start
	iCountRx++;
```

```
        }
}

// Azimuth : Motor Pwm + L/R
void PSetMotorAz(void)
{
	if (iTimeAz < iSensor[0])				// Left ?
	{
		iDeltaAz = iSensor[0] - iTimeAz;
		if (iDeltaAz < (kAkurasiAz+kDeadZone))		// Move to Stop
		{
			if (iDeltaAz < kAkurasiAz)
			{
				cStatusAz = 0x00;		// Stop
			}
			else if (!cStatusAz)			// Dead zone
			{
				cStatusAz = 0x01;		// Left
			}
		}
		else
		{
			cStatusAz = 0x01;			// Left
		}
	}
	else

	{
		iDeltaAz = iTimeAz - iSensor[0];
		if (iDeltaAz < (kAkurasiAz+kDeadZone))		// Move to Stop
		{
			if (iDeltaAz < kAkurasiAz)
			{
				cStatusAz = 0x00;		// Stop
			}
			else if (!cStatusAz)			// Dead zone
			{
				cStatusAz = 0x02;		// Right
			}
		}
		else
		{
			cStatusAz = 0x02;			// Right
		}
	}

	if (cStatusAz==0x01)					// Left
	{
		cTimeAz = kPwmAz;				// Pwm : 35%
		cPortC |= 0x01;					// C0 High
		cPortC &= 0xFD;					// C1 Low
	}
	else if (cStatusAz==0x02)				// Right
	{
		cTimeAz = kPwmAz;				// Pwm : 35%
		cPortC &= 0xFE;					// C0 Low
```

```
			cPortC |= 0x02;				// C1 High
		}
		else							// Stop
		{
			cPortC &= 0xFB;				// C2 Low
			cPortC |= 0x03;				// C0+C1 High
			cTimeAz = 0x00;
		}
}

// Elevasi : Motor Pwm + L/R
void PSetMotorEl(void)
{
//		cStatusEl = 0x00;
//		Elevasi : Dn or Up
		if (iTimeEl < iSensor[1])			// Dn ?
		{
			iDeltaEl = iSensor[1] - iTimeEl;
			if (iDeltaEl < (kAkurasiEl+kDeadZone))	// Move to Stop
			{
				if (iDeltaEl < kAkurasiEl)
				{
					cStatusEl = 0x00;	// Stop
				}
				else if (!cStatusEl)		// Dead zone
				{
					cStatusEl = 0x01;	// Dn
				}
			}
			else
			{
				cStatusEl = 0x01;		// Dn
			}
		}
		else

		{
			iDeltaEl = iTimeEl - iSensor[1];
			if (iDeltaEl < (kAkurasiEl+kDeadZone))	// Move to Stop
			{
				if (iDeltaEl < kAkurasiEl)
				{
					cStatusEl = 0x00;	// Stop
				}
				else if (!cStatusEl)		// Dead zone
				{
					cStatusEl = 0x02;	// Up
				}
			}
			else
			{
				cStatusEl = 0x02;		// Up
			}
		}

		if (cStatusEl==0x01)				// Dn
		{
```

```
		cTimeEl = kPwmEl;			// Pwm : 35%
		cPortC |= 0x20;				// C5 High
		cPortC &= 0xBF;				// C6 Low
	}
	else if (cStatusEl==0x02)			// Up
	{
		cTimeEl = kPwmEl;			// Pwm : 35%
		cPortC &= 0xDF;				// C5 Low
		cPortC |= 0x40;				// C6 High
	}
	else						// Stop
	{
		cPortC &= 0x7F;				// C7 Low
		cPortC |= 0x60;				// C5+C6 High
		cTimeEl = 0x00;
	}
}

// LED : On + Off by PWM
void PSetInput(void)
{
	// Data Az+El : PC atau Input Potensiometer
	if (bPcOk)
	{
		iTimeAz = iPcInAz;
		iTimeEl = iPcInEl;
	}
	else
	{
		iTimeAz = iSensor[2];
		iTimeEl = iSensor[3];
	}
}

// ------------------------------------------------------------------------------------------------------------------
// Fungsi : Baca Data ADC di PORT A
void PDataAdc()
{
	if (ADCSRA & 0x10)		// Check bila proses konversi komplit
	{
		// ADC : Baca Adc channel terakhir
		ADCSRA |= 0x10;
		cAdcL = ADCL;				// Baca ADC Low
		cAdcH = ADCH;				// Baca ADC High
		iSensor[iAdcNo] = ADCW;			// Baca ADC Word

		iAdcNo++;
		if (iAdcNo >= kMaxAdc)
		{
			iAdcNo = 0;		// Jika pin terakhir kembali ke pin awal
		}
		ADMUX = iAdcNo | ADC_REF;		// Baca Pin berikutnya
		ADCSRA |= 0x40;

		if (iAdcNo == 0)
		{
			PSetInput();
```

```
                    PSetMotorAz();
                    PSetMotorEl();
                    PORTC = cPortC;
                }
        }
}

// Fungsi : LOOP Forever
//--------------------------------------------------------------------------------------------------------------------
void PLooping(void)
{
//      PORTB = cDataC;
        ADCSRA |= 0x40;

//      while(1)
        for(;;)
        {
//              PORTB = PINA;

                // Proses 0 : Baca Data Switch
                cPortB = PINB;
                if (cPortB & 0x01)                                  // check B0 High
                {
                        bPcOk = 0x00;                               // Data from Manual
                }
                else
                {
                        bPcOk = 0x01;                               // Data from PC
                }

                // Proses 1 : Baca Data ADC
                PDataAdc();                                         // Baca Data ADC

                // Proses 2: COMM : Kirim Data via USART if Necessary
                if (cTimeTx > 2)                                    // 60/3 = 20 Hz
                //if (cTimeTx > 4)                                  // 60/5 = 12 Hz
                //if (cTimeTx > 5)                                  // 60/6 = 10 Hz
                {

                cTimeTx = 0;
                iTxNo = 0;
                cByteTx[0] = kHH;
                cByteTx[1] = 0x01;                                  // Ack : uC to PC
                cByteTx[2] = (iSensor[0] >> 8) & 0xFF;              // Sensor AZ : High
                cByteTx[3] = iSensor[0] & 0xFF;
                cByteTx[4] = (iSensor[1] >> 8) & 0xFF;              // Sensor EL : High
                cByteTx[5] = iSensor[1] & 0xFF;
                cByteTx[6] = (iSensor[2] >> 8) & 0xFF;              // Manual AZ : High
                cByteTx[7] = iSensor[2] & 0xFF;
                cByteTx[8] = (iSensor[3] >> 8) & 0xFF;              // Manual EL : High
                cByteTx[9] = iSensor[3] & 0xFF;
                cByteTx[10] = (iPcInAz >> 8) & 0xFF;                // Komp AZ : High
                cByteTx[11] = iPcInAz & 0xFF;
                cByteTx[12] = (iPcInEl >> 8) & 0xFF;                // Komp EL : High
                cByteTx[13] = iPcInEl & 0xFF;
                cByteTx[14] = cStatusAz | (cStatusEl << 4);         // Status : El+Az
                cByteTx[15] = kTT;
```

```
				}

				// Proses 3 : Tx Ready
				if (iTxNo < kMaxTx)
				{
					PDataTx();
				}

				// Proses 4: COMM : Check Comm dan Terima Data
				if (UCSRA & (1<<RXC))			// Menunggu Rx Flag komplit
				{
					PDataRx();
				}
		}
}


// Fungsi : UTAMA
//----------------------------------------------------------------------------------------------------------------------------
int main(void)
{
		PInitHW();						// Initialisasi : Hardware
		PInitData();					// Initialisasi : Data
		sei();							// Global Interrupt : Enabled
		PLooping();						// Loop : FOREVER
		return 0;
}
```

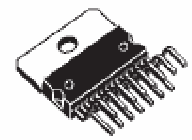
LAMPIRAN C

DATA SHEET

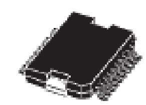
L298

DUAL FULL-BRIDGE DRIVER

- OPERATING SUPPLY VOLTAGE UP TO 46 V
- TOTAL DC CURRENT UP TO 4 A
- LOW SATURATION VOLTAGE
- OVERTEMPERATURE PROTECTION
- LOGICAL "0" INPUT VOLTAGE UP TO 1.5 V (HIGH NOISE IMMUNITY)



Multiwatt15



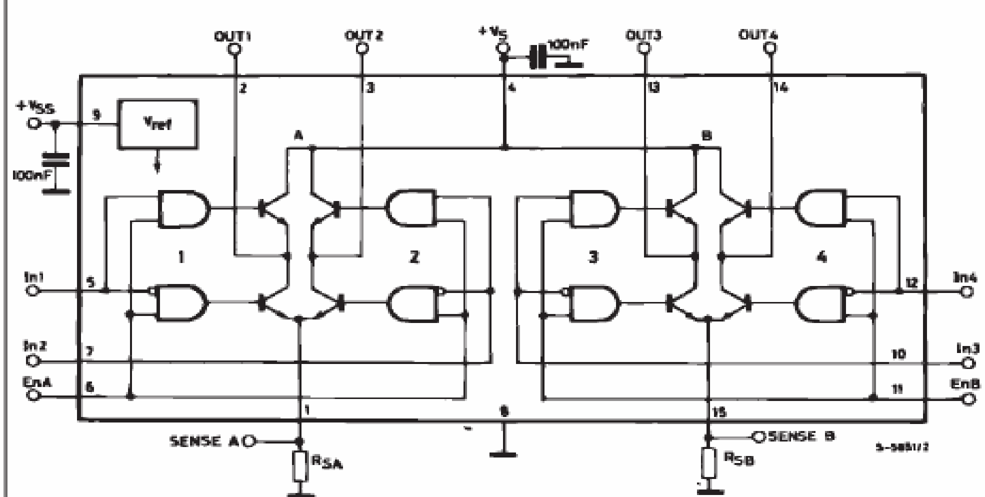
PowerSO20

ORDERING NUMBERS : L298N (Multiwatt Vert.)
L298HN (Multiwatt Horiz.)
L298P (PowerSO20)

DESCRIPTION

The L298 is an integrated monolithic circuit in a 15-lead Multiwatt and PowerSO20 packages. It is a high voltage, high current dual full-bridge driver designed to accept standard TTL logic levels and drive inductive loads such as relays, solenoids, DC and stepping motors. Two enable inputs are provided to enable or disable the device independently of the input signals. The emitters of the lower transistors of each bridge are connected together and the corresponding external terminal can be used for the connection of an external sensing resistor. An additional supply input is provided so that the logic works at a lower voltage.

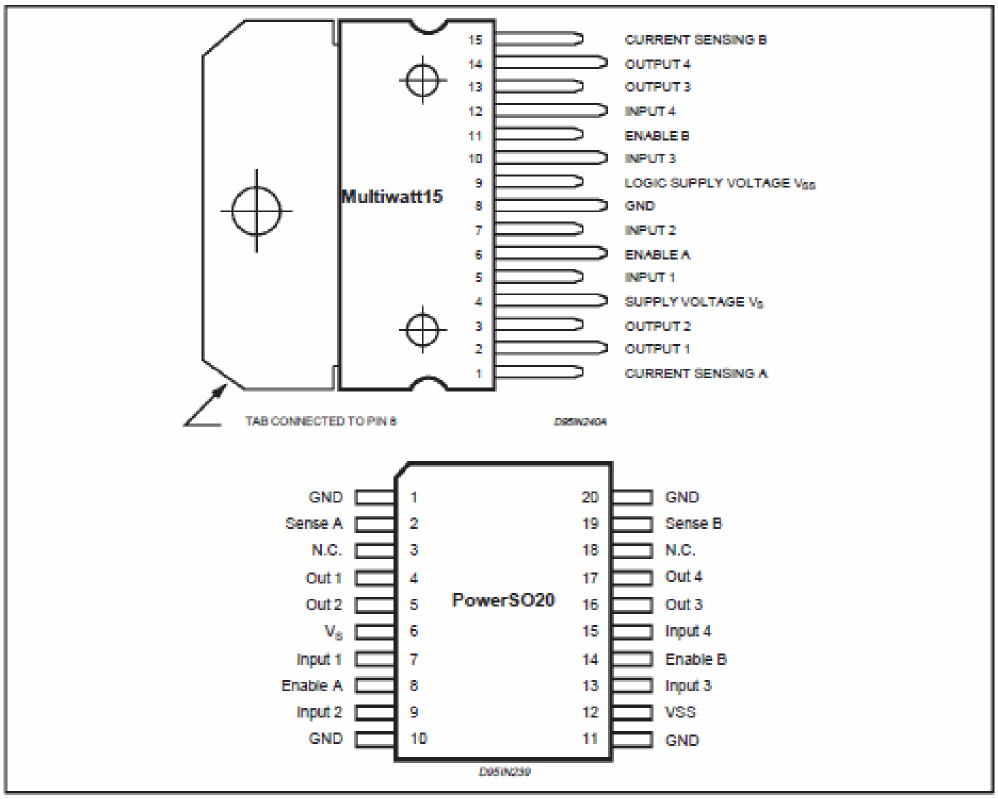
BLOCK DIAGRAM



ABSOLUTE MAXIMUM RATINGS

Symbol	Parameter	Value	Unit
V_S	Power Supply	50	V
V_{SS}	Logic Supply Voltage	7	V
V_I,V_{en}	Input and Enable Voltage	–0.3 to 7	V
I_O	Peak Output Current (each Channel) – Non Repetitive (t = 100µs) –Repetitive (80% on –20% off; t_{on} = 10ms) –DC Operation	 3 2.5 2	 A A A
V_{sens}	Sensing Voltage	–1 to 2.3	V
P_{tot}	Total Power Dissipation (T_{case} = 75°C)	25	W
T_{op}	Junction Operating Temperature	–25 to 130	°C
T_{stg}, T_j	Storage and Junction Temperature	–40 to 150	°C

PIN CONNECTIONS (top view)



THERMAL DATA

Symbol	Parameter		PowerSO20	Multiwatt15	Unit
$R_{th\ j\text{-}case}$	Thermal Resistance Junction-case	Max.	–	3	°C/W
$R_{th\ j\text{-}amb}$	Thermal Resistance Junction-ambient	Max.	13 (*)	35	°C/W

(*) Mounted on aluminum substrate

PIN FUNCTIONS (refer to the block diagram)

MW.15	PowerSO	Name	Function
1;15	2;19	Sense A; Sense B	Between this pin and ground is connected the sense resistor to control the current of the load.
2;3	4;5	Out 1; Out 2	Outputs of the Bridge A; the current that flows through the load connected between these two pins is monitored at pin 1.
4	6	V_S	Supply Voltage for the Power Output Stages. A non-inductive 100nF capacitor must be connected between this pin and ground.
5;7	7;9	Input 1; Input 2	TTL Compatible Inputs of the Bridge A.
6;11	8;14	Enable A; Enable B	TTL Compatible Enable Input: the L state disables the bridge A (enable A) and/or the bridge B (enable B).
8	1,10,11,20	GND	Ground.
9	12	VSS	Supply Voltage for the Logic Blocks. A100nF capacitor must be connected between this pin and ground.
10; 12	13;15	Input 3; Input 4	TTL Compatible Inputs of the Bridge B.
13; 14	16;17	Out 3; Out 4	Outputs of the Bridge B. The current that flows through the load connected between these two pins is monitored at pin 15.
–	3;18	N.C.	Not Connected

ELECTRICAL CHARACTERISTICS (V_S = 42V; V_{SS} = 5V, T_j = 25°C; unless otherwise specified)

Symbol	Parameter	Test Conditions	Min.	Typ.	Max.	Unit
V_S	Supply Voltage (pin 4)	Operative Condition	V_{IH} +2.5		46	V
V_{SS}	Logic Supply Voltage (pin 9)		4.5	5	7	V
I_S	Quiescent Supply Current (pin 4)	V_{en} = H; I_L = 0 V_i = L V_i = H		13 50	22 70	mA mA
		V_{en} = L V_i = X			4	mA
I_{SS}	Quiescent Current from V_{SS} (pin 9)	V_{en} = H; I_L = 0 V_i = L V_i = H		24 7	36 12	mA mA
		V_{en} = L V_i = X			6	mA
V_{iL}	Input Low Voltage (pins 5, 7, 10, 12)		–0.3		1.5	V
V_{iH}	Input High Voltage (pins 5, 7, 10, 12)		2.3		VSS	V
I_{iL}	Low Voltage Input Current (pins 5, 7, 10, 12)	V_i = L			–10	µA
I_{iH}	High Voltage Input Current (pins 5, 7, 10, 12)	Vi = H ≤ V_{SS} –0.6V		30	100	µA
V_{en} = L	Enable Low Voltage (pins 6, 11)		–0.3		1.5	V
V_{en} = H	Enable High Voltage (pins 6, 11)		2.3		V_{SS}	V
I_{en} = L	Low Voltage Enable Current (pins 6, 11)	V_{en} = L			–10	µA
I_{en} = H	High Voltage Enable Current (pins 6, 11)	V_{en} = H ≤ V_{SS} –0.6V		30	100	µA
$V_{CEsat(H)}$	Source Saturation Voltage	I_L = 1A I_L = 2A	0.95	1.35 2	1.7 2.7	V V
$V_{CEsat(L)}$	Sink Saturation Voltage	I_L = 1A (5) I_L = 2A (5)	0.85	1.2 1.7	1.6 2.3	V V
V_{CEsat}	Total Drop	I_L = 1A (5) I_L = 2A (5)	1.80		3.2 4.9	V V
V_{sens}	Sensing Voltage (pins 1, 15)		–1 (1)		2	V

ELECTRICAL CHARACTERISTICS (continued)

Symbol	Parameter	Test Conditions	Min.	Typ.	Max.	Unit
T_1 (V_i)	Source Current Turn-off Delay	0.5 V_i to 0.9 I_L (2); (4)		1.5		μs
T_2 (V_i)	Source Current Fall Time	0.9 I_L to 0.1 I_L (2); (4)		0.2		μs
T_3 (V_i)	Source Current Turn-on Delay	0.5 V_i to 0.1 I_L (2); (4)		2		μs
T_4 (V_i)	Source Current Rise Time	0.1 I_L to 0.9 I_L (2); (4)		0.7		μs
T_5 (V_i)	Sink Current Turn-off Delay	0.5 V_i to 0.9 I_L (3); (4)		0.7		μs
T_6 (V_i)	Sink Current Fall Time	0.9 I_L to 0.1 I_L (3); (4)		0.25		μs
T_7 (V_i)	Sink Current Turn-on Delay	0.5 V_i to 0.9 I_L (3); (4)		1.6		μs
T_8 (V_i)	Sink Current Rise Time	0.1 I_L to 0.9 I_L (3); (4)		0.2		μs
fc (V_i)	Commutation Frequency	I_L = 2A		25	40	KHz
T_1 (V_{en})	Source Current Turn-off Delay	0.5 V_{en} to 0.9 I_L (2); (4)		3		μs
T_2 (V_{en})	Source Current Fall Time	0.9 I_L to 0.1 I_L (2); (4)		1		μs
T_3 (V_{en})	Source Current Turn-on Delay	0.5 V_{en} to 0.1 I_L (2); (4)		0.3		μs
T_4 (V_{en})	Source Current Rise Time	0.1 I_L to 0.9 I_L (2); (4)		0.4		μs
T_5 (V_{en})	Sink Current Turn-off Delay	0.5 V_{en} to 0.9 I_L (3); (4)		2.2		μs
T_6 (V_{en})	Sink Current Fall Time	0.9 I_L to 0.1 I_L (3); (4)		0.35		μs
T_7 (V_{en})	Sink Current Turn-on Delay	0.5 V_{en} to 0.9 I_L (3); (4)		0.25		μs
T_8 (V_{en})	Sink Current Rise Time	0.1 I_L to 0.9 I_L (3); (4)		0.1		μs

1) 1)Sensing voltage can be –1 V for t ≤ 50 μsec; in steady state V_{sens} min ≥ –0.5 V.
2) See fig. 2.
3) See fig. 4.
4) The load must be a pure resistor.

Figure 1 : Typical Saturation Voltage vs. Output Current.

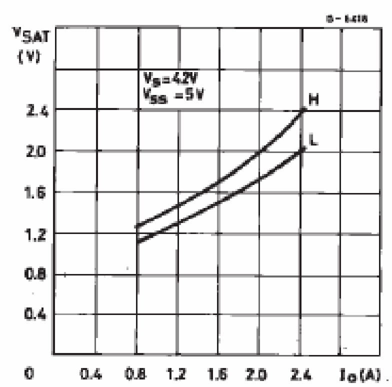
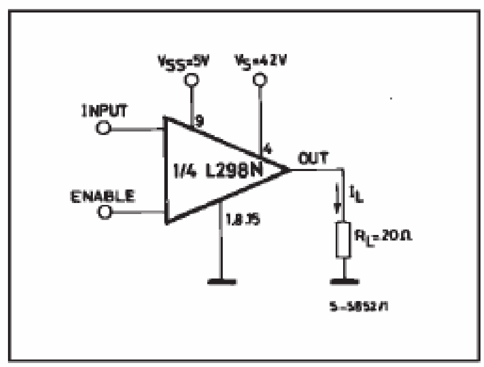


Figure 2 : Switching Times Test Circuits.



Note : For INPUT Switching, set EN = H
For ENABLE Switching, set IN = H

Figure 3 : Source Current Delay Times vs. Input or Enable Switching.

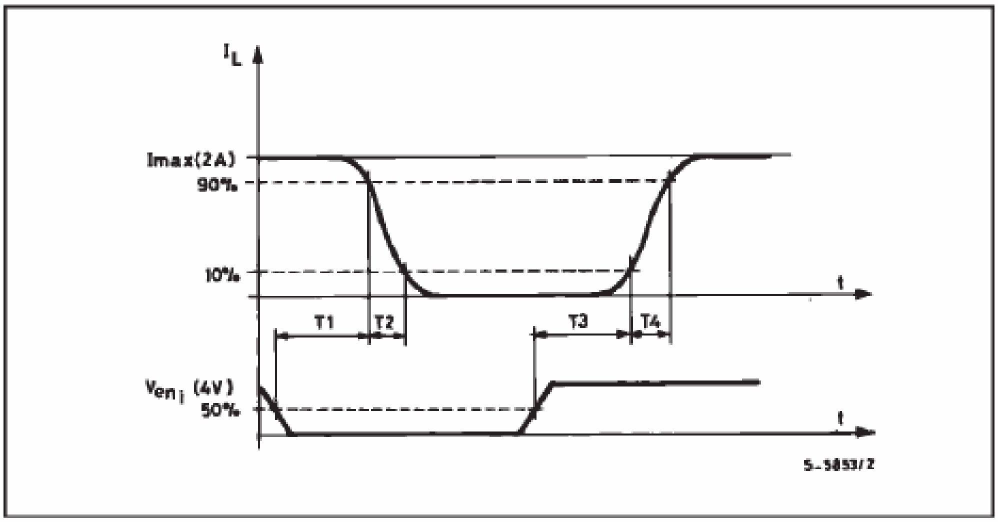
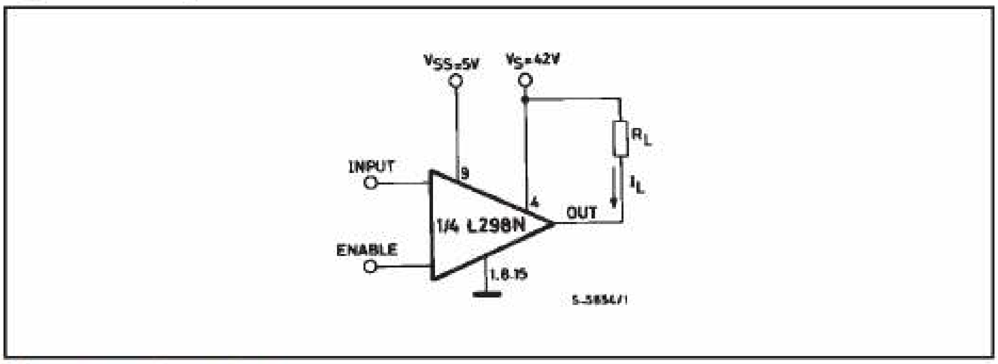


Figure 4 : Switching Times Test Circuits.



Note : For INPUT Switching, set EN = H
For ENABLE Switching, set IN = L

Figure 5 : Sink Current Delay Times vs. Input 0 V Enable Switching.

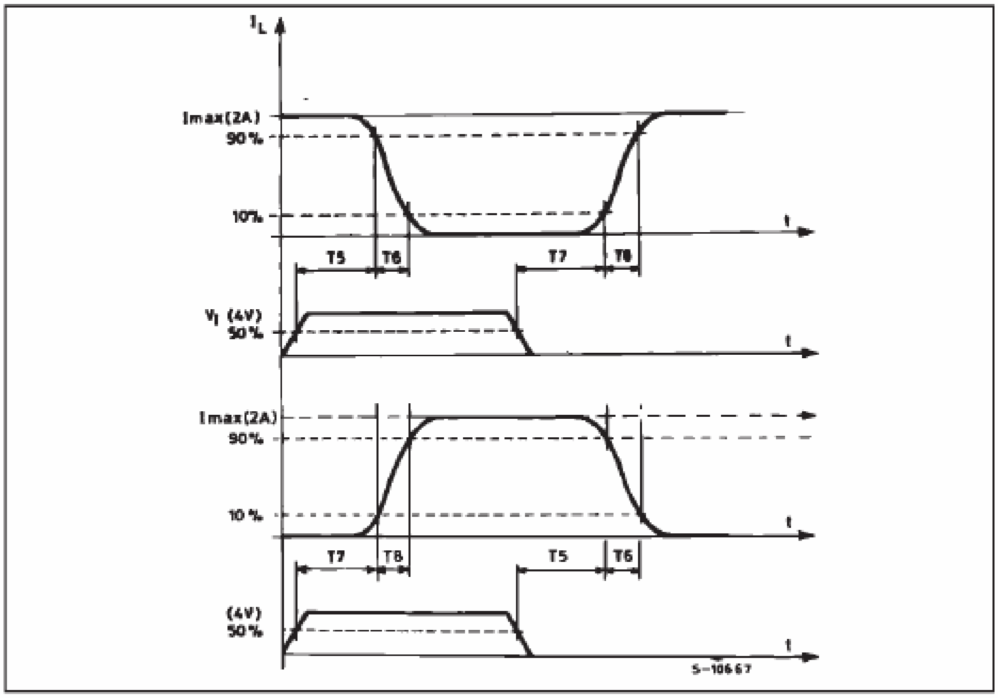
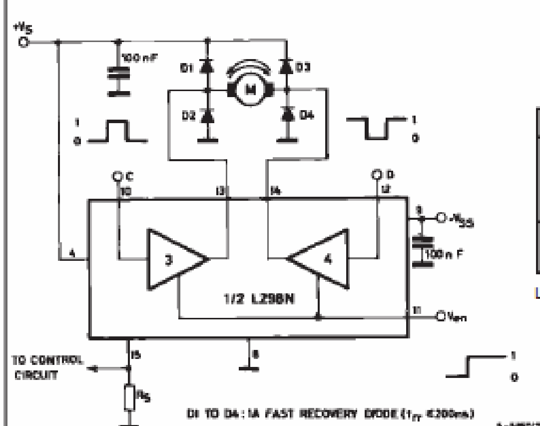


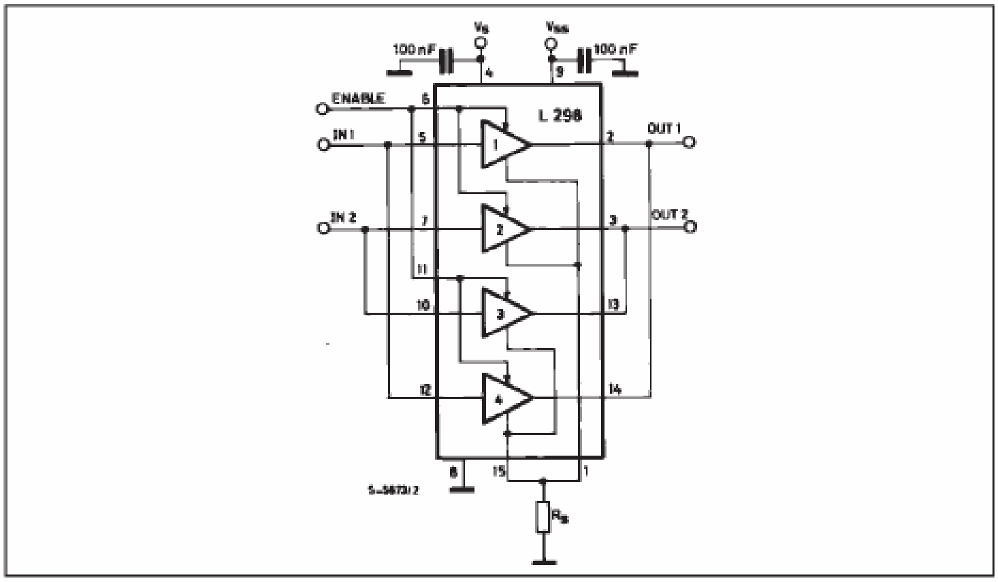
Figure 6 : Bidirectional DC Motor Control.



Inputs		Function
V_{en} = H	C = H ; D = L	Forward
	C = L ; D = H	Reverse
	C = D	Fast Motor Stop
V_{en} = L	C = X ; D = X	Free Running Motor Stop

L = Low H = High X = Don't care

Figure 7 : For higher currents, outputs can be paralleled. Take care to parallel channel 1 with channel 4 and channel 2 with channel 3.



APPLICATION INFORMATION (Refer to the block diagram)

1.1. POWER OUTPUT STAGE

The L298 integrates two power output stages (A; B). The power output stage is a bridge configuration and its outputs can drive an inductive load in common or differenzial mode, depending on the state of the inputs. The current that flows through the load comes out from the bridge at the sense output : an external resistor (R_{SA} ; R_{SB}.) allows to detect the intensity of this current.

1.2. INPUT STAGE

Each bridge is driven by means of four gates the input of which are In1 ; In2 ; EnA and In3 ; In4 ; EnB. The In inputs set the bridge state when The En input is high ; a low state of the En input inhibits the bridge. All the inputs are TTL compatible.

2. SUGGESTIONS

A non inductive capacitor, usually of 100 nF, must be foreseen between both Vs and Vss, to ground, as near as possible to GND pin. When the large capacitor of the power supply is too far from the IC, a second smaller one must be foreseen near the L298.

The sense resistor, not of a wire wound type, must be grounded near the negative pole of Vs that must be near the GND pin of the I.C.

Each input must be connected to the source of the driving signals by means of a very short path.

Turn-On and Turn-Off : Before to Turn-ON the Supply Voltage and before to Turn it OFF, the Enable input must be driven to the Low state.

3. APPLICATIONS

Fig 6 shows a bidirectional DC motor control Schematic Diagram for which only one bridge is needed. The external bridge of diodes D1 to D4 is made by four fast recovery elements ($t_{rr} \leq$ 200 nsec) that must be chosen of a VF as low as possible at the worst case of the load current.

The sense output voltage can be used to control the current amplitude by chopping the inputs, or to provide overcurrent protection by switching low the enable input.

The brake function (Fast motor stop) requires that the Absolute Maximum Rating of 2 Amps must never be overcome.

When the repetitive peak current needed from the load is higher than 2 Amps, a paralleled configuration can be chosen (See Fig.7).

An external bridge of diodes are required when inductive loads are driven and when the inputs of the IC are chopped; Shottky diodes would be preferred.

This solution can drive until 3 Amps In DC operation and until 3.5 Amps of a repetitive peak current.

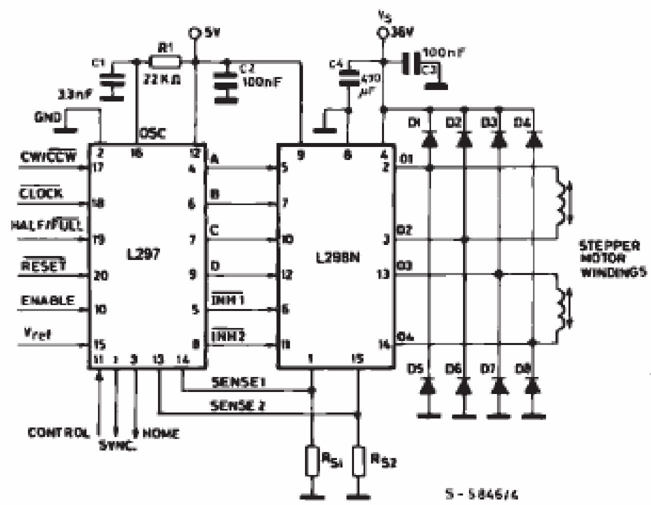
On Fig 8 it is shown the driving of a two phase bipolar stepper motor ; the needed signals to drive the inputs of the L298 are generated, in this example, from the IC L297.

Fig 9 shows an example of P.C.B. designed for the application of Fig 8.

Fig 10 shows a second two phase bipolar stepper motor control circuit where the current is controlled by the I.C. L6506.

Figure 8 : Two Phase Bipolar Stepper Motor Circuit.

This circuit drives bipolar stepper motors with winding currents up to 2 A. The diodes are fast 2 A types.

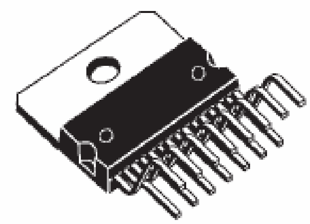


$R_{S1} = R_{S2} = 0.5\ \Omega$

D1 to D8 = 2 A Fast diodes { $V_F \leq 1.2$ V @ I = 2 A; trr ≤ 200 ns

DIM.	mm			inch		
	MIN.	TYP.	MAX.	MIN.	TYP.	MAX.
A			5			0.197
B			2.65			0.104
C			1.6			0.063
D		1			0.039	
E	0.49		0.55	0.019		0.022
F	0.66		0.75	0.026		0.030
G	1.02	1.27	1.52	0.040	0.050	0.060
G1	17.53	17.78	18.03	0.690	0.700	0.710
H1	19.6			0.772		
H2			20.2			0.795
L	21.9	22.2	22.5	0.862	0.874	0.886
L1	21.7	22.1	22.5	0.854	0.870	0.886
L2	17.65		18.1	0.695		0.713
L3	17.25	17.5	17.75	0.679	0.689	0.699
L4	10.3	10.7	10.9	0.406	0.421	0.429
L7	2.65		2.9	0.104		0.114
M	4.25	4.55	4.85	0.167	0.179	0.191
M1	4.63	5.08	5.53	0.182	0.200	0.218
S	1.9		2.6	0.075		0.102
S1	1.9		2.6	0.075		0.102
Dia1	3.65		3.85	0.144		0.152

OUTLINE AND MECHANICAL DATA



Multiwatt15 V