LENTARK®

PWM-CORE
PP504F0A-02W30

FEATURES

- Programmable 3 Pwm Output: P1, P2, P3
- Accurate Phase, Frequency and Duty Cycle.
- Frequency Range: 4 Hz – 500 KHz
  - Frequency Tolerance (Max): ±0.2%
  - Frequency Stability (80ºC): 100 ppm
- Tunable Period Range: 250.00ms – 120.00s
- Phase Range: 0.00º – 360.00º
- Duty Cycle Range: 0.00% – 100.00%
- Easy communication: UART (Rx, Tx)
- Ability to Autosave Data to Internal Memory.
- Wide Input Supply Voltage Range: 5V – 24V
- Low Output Voltage Swing: 0 – 125 µV

APPLICATIONS

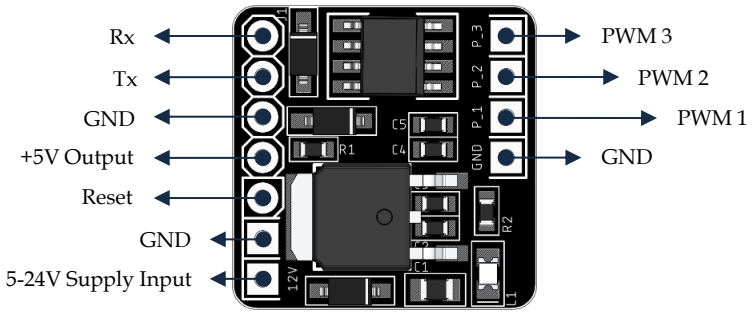
- P & N Channel MOSFET, IGBT Switching.
- H Bridge & Half Bridge Drivers.
- Switching Amplifiers.
- 3-Phase Applications.
- Motor Control.
- R, L and/or C Load Control.
- Hobby.

GENERAL DESCRIPTION

PP504F0A-02W30 is a PWM core with 5V – 24V input voltage and three channel signal output. **The frequency, phase and duty cycle characteristics of all channels can be defined separately**. Signal outputs have amplitudes of 5V. It has a maximum tolerance of ± 0.2% in the 4 Hz – 500 KHz band and also has the ability to be calibrated for operations in lower frequency bands.

The communication is quite convenient and simple. All control operations are provided with the codes to be sent to the Rx pin of PP504F0A-02W30. The **Protocols** are shown together with the examples below. The device keeps all signal configurations in memory. Thus, it does not need reconfiguration in every use.

PINOUTS



ELECTRICAL SPECIFICATIONS

∏ Pushing the device to operate above the "Max." listed in the table below may cause the device to overheat and to take up permanent damage. It is inconclusive that the device will function beyond the operating limits as set out in this technical document. Prolonged exposure to work under "maximum" rating conditions may affect device reliability.

Table 1: *Electrical Specifications.*

Conditions: Unless Otherwise Noted, $T_O = +25^oC$ and $9V \leq V_{IN} \leq 18V$.						
Parameters	**Sym**	**Min**	**Typ**	**Max**	**Units**	**Condition**
Input						
Input Voltage	V_{IN}	5	12	24	V	DC
Input Current [No Load]	I_{IN}	5.26 6.51	9.29 11.23	9.48 11.37	mA	$f = 1\ \ Hz$ $f = 500\ KHZ$
Output						
Output Voltage, High	$V_{OUT,HIGH}$	4.200	5	5.100	V	$V_{IN} = 12V$
Output Voltage, Low	$V_{OUT,LOW}$	0	0	0.615	V	
Output Resistance, High	$R_{OUT,HIGH}$	–	–	200	Ω	
Output Resistance, Low	$R_{OUT,LOW}$	75	–	–	Ω	
Total Output Current*	I_{OT}	–	30	45	mA	
Switching						
Rise Time	t_R	–	15	32	ns	C_L = 0 pF
			30	65		C_L = 50 pF
Fall Time	t_F	–	15	30	ns	C_L = 0 pF
			30	60		C_L = 50 pF
Output Power Dissipation	W_{PD}	–	100	720	mW	**Note 1**

* **:** It refers to the total current that can be drawn from the signal outputs.

Note1 : It refers to the power that can be consumed in the sum of the signal outputs.

Tolerance & Sensitivity						
Parameters	**Sym**	**Min**	**Typ**	**Max**	**Units**	**Condition**
Frequency Tolerance *		–	0.05	0.2	% [percent]	250 ms – 2 µs
		–	–	5		120 s – 250 ms
Frequency Sensitivity		–	0.0312	–	µs	$T > 2$ µs
Duty Cycle Sensitivity (0.00 – 100.00)	D	0.01	–	0.01	% [percent]	$T > 312$ µs
		0.01	–	0.1		$T > 31$ µs
		0.1	–	1		$T < 31$ µs
Phase Angle Sensitivity (0.00 – 360.00)	P	0.01	–	0.01	o [degrees]	$T > 1.12$ ms
		0.01	–	0.1		$T > 112$ us
		0.1	–	1		$T < 112$ us

* **:** The period between 120 s - 250 ms can be calibrated manually. See "**Fast Examples of Protocols**" to see examples.

PIN DESCRIPTIONS

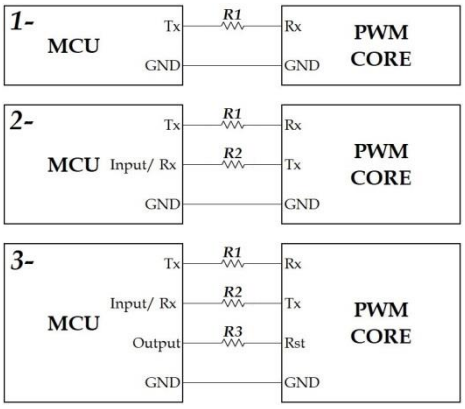
Pin	*Description*	*Notes*	*Types of Connections*
Rx	Used to read data from any MCU. (9600 Baud Rate – 8 Bit Buffer)	It connects to the Tx port of any MCU.	
Tx	This pin informs whether Pwm-Core's Rx pin is ready to read new data. (1: Ready – 0: Busy)	It connects to the Rx or Input port of any MCU. ($0 \leq R_{1,2,3} \leq 470\ \Omega$)	
Rst	Pwm-Core's reset pin. (1: Active – 0: Passive)	It can be connected to the Output pin of any MCU, and can be used as Pwm-Core's reset or enable pin.	
+5V	5V Output	Max. 130 mA can be drain.	
12V	5-24V Supply Voltage	Nominal 12V	
GND	GND	–	
PWM1	Signal Output 1	For detailed information, see also **Electrical Characteristics** table.	
PWM2	Signal Output 2		
PWM3	Signal Output 3		

Figure 1: *Typical Connection Schemes.*

Table 2: *Pin Descriptions.*

The device operates PWM channels synchronously. This allows assignment of phase angle between channels. The device processes a single command that commands a parameter of any PWM channel, records all values and restarts all PWM channels synchronously with the new values.

PROTOCOLS

The connections of the Pwm-Core should be done as specified in the **Pin Descriptions** title. The UART communication protocol of the MCU that will send the command must be configured according to the information given in the **Protocol Information** header and the String / Char * / Char [] variable must be prepared in accordance with the following two rules for the command to be sent.

Note: The device keeps all signal configurations in memory. Thus, it does not need reconfiguration in every use.

Rule 1 – Format:

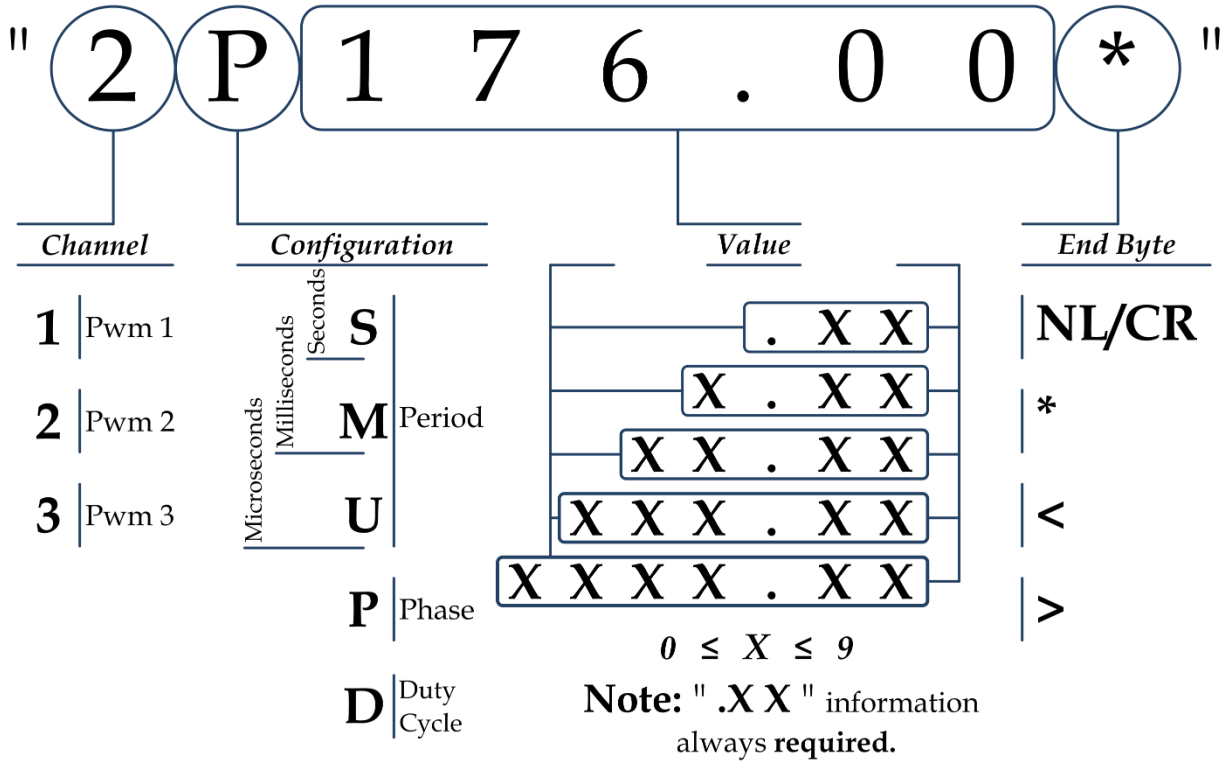


Figure 2: *Format.*

End Bytes	Explanation	Examples
'*' or NL/CR	**The command completion byte.** If a command array is uninitialized, the command is processed, all values are saved and all PWM channels are restarted synchronously.	1P1.23* 2D74.11*
'<'	**The command array initiation byte.** All commands are processed but not transmitted to PWM channels until the array termination byte arrives.	1M1.00< 2P120.00* … 2D90.00* 3D12.05>
'>'	**The command array termination byte**. If there is an initiated array, it completes array, if not, it completes the singular command. It processes all commands into PWM channels, saves all values and restarts all PWM channels synchronously.	

Table 3: *Explanation of End Bytes.*

Rule 2 – Protocol, Processing Time and Limits:

Protocol Information:

Mode : Asynchronous
Baud Rate : 9600
Data Polarity : Active-High
Rx Read Bits : 8 Bits

Limits:

	Lowest Value	Highest Value
S	0.50	120.00
M	0.01	500.00
U	2.00	2000.00
P	0.00	360.00
D	0.00	100.00

Processing_Time:

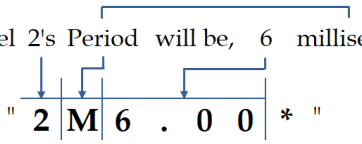
If Pwm-Core's Tx[Rx_Rdy] pin is not to be used, Pwm-Core's processing time must be taken into consideration in order to must be kept waiting while the first data is being processed. The processing times specified in **Table 4** show the highest and typical processing times determined. However, since each configuration goes through a different set of mathematical operations, it cannot be guaranteed that it will not exceed the specified processing times.

Configuration	*Typical*	*Max.*
Period	4.260 ms	4.875 ms
Phase	3.625 ms	4.120 ms
Duty Cycle	3.785 ms	4.215 ms

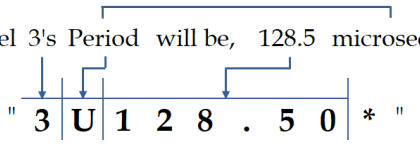
Table 4: *Processing Times.*

Fast Examples of Protocols:

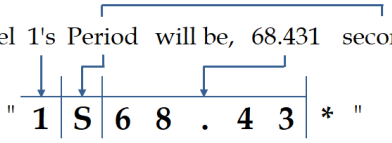
∏ If Channel 2's Period will be, 6 milliseconds;



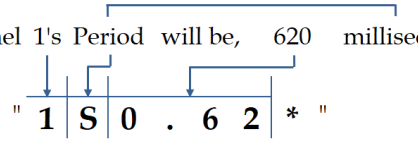
∏ If Channel 3's Period will be, 128.5 microseconds;



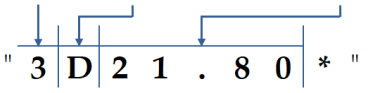
∏ If Channel 1's Period will be, 68.431 seconds;



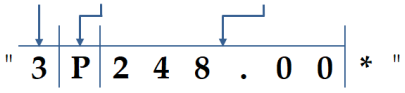
∏ If Channel 1's Period will be, 620 milliseconds;



∏ If Channel 3's Duty Cycle will be 21.8%;



∏ If Channel 3's Phase will be 248°;

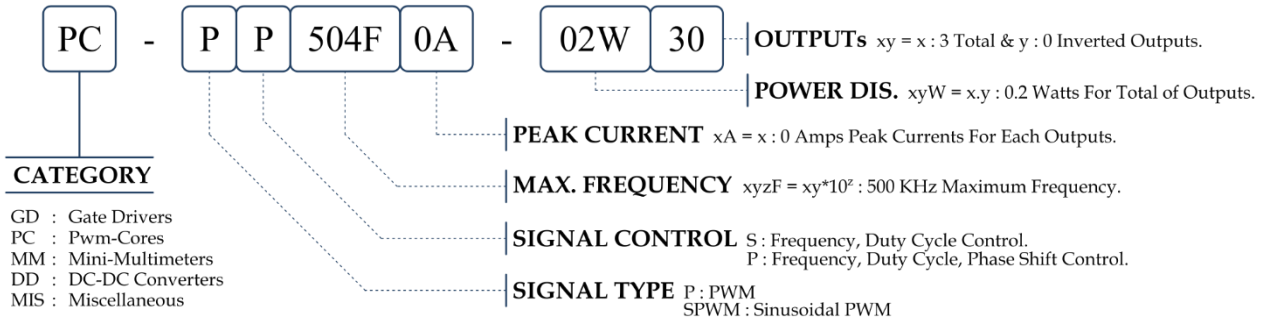


Examples For Coding:

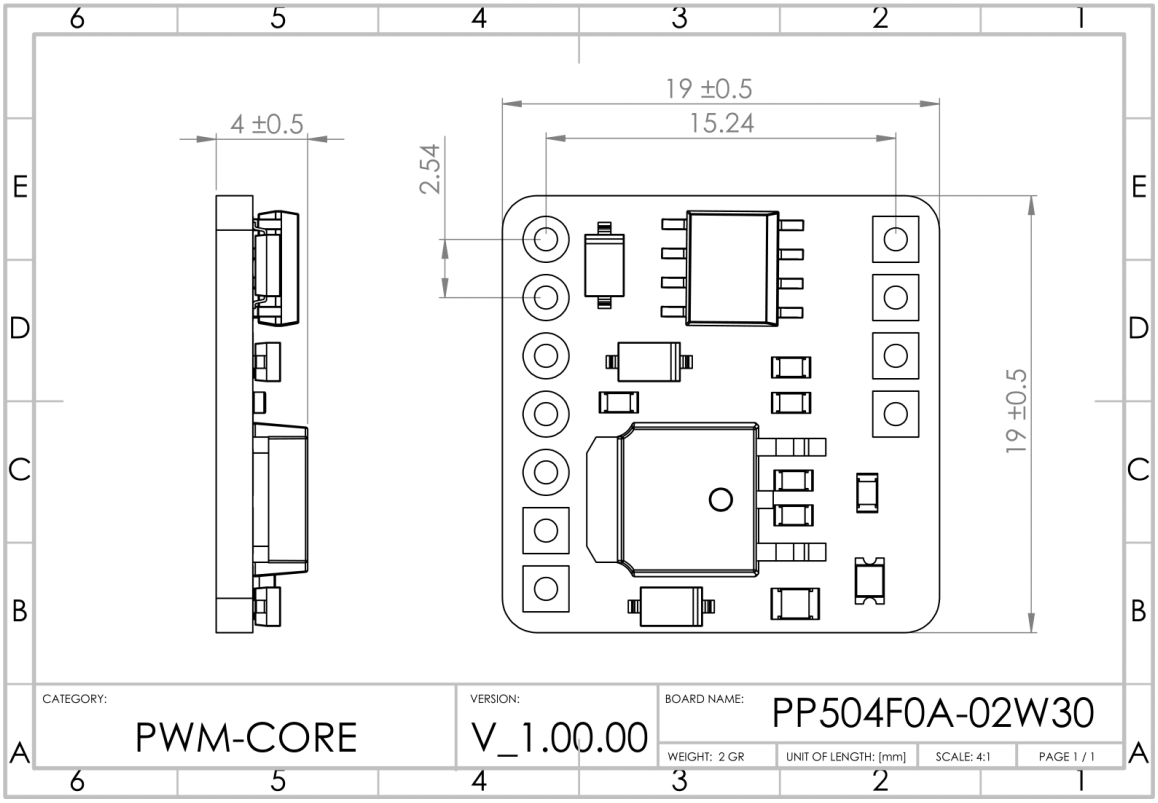
```
//PP504F0A-02W30
#define Rx_Ready 10 //Any Input Pin
#include <SoftwareSerial.h>
SoftwareSerial _mySerial(10, 11);// RX, TX

//----------Baud Rate-------------------
//Baud Rate: 4 -> 9600 | 3 -> 19200 | 2 -> 57600 | 1 -> 115200
uint16_t cong_Baud_[2]= {4, 0};
//-----------Period Type-------------------------
// S: Seconds | M: Milliseconds | U: Microseconds
uint8_t channel_1_SMU_= 'U';
uint8_t channel_2_SMU_= 'M';
uint8_t channel_3_SMU_= 'M';
//------------Period-------------------------------
//{Integer Part, Floating Part}
//Exp: {245,75} => if _config == 'U' than Period = 245.75 microseconds
uint16_t channel_1_period_[2]= {500, 5};
uint16_t channel_2_period_[2]= {10, 0};
uint16_t channel_3_period_[2]= {15, 38};
//-------------Phase-------------------------------
uint16_t channel_1_phase_[2]= {0, 0};
uint16_t channel_2_phase_[2]= {120, 50};
uint16_t channel_3_phase_[2]= {240, 0};
//-----------Duty Cycle---------------------------
uint16_t channel_1_duty_[2]= {25, 0};
uint16_t channel_2_duty_[2]= {25, 17};
uint16_t channel_3_duty_[2]= {50, 0};


void setup() {
Serial.begin(9600); while (!Serial) {;}
_mySerial.begin(9600); // Set Baud Rate
pinMode(Rx_Ready, INPUT_PULLUP); // Set Rx_Ready to input & Turn on pull-up resistors
//send_configuration(1, 'B', cong_Baud_, '>');
send_configuration(1, channel_1_SMU_, channel_1_period_, '<');
send_configuration(2, channel_2_SMU_, channel_2_period_, '*');
send_configuration(3, channel_3_SMU_, channel_3_period_, '*');
send_configuration(1, 'P', channel_1_phase_, '*');
send_configuration(2, 'P', channel_2_phase_, '*');
send_configuration(3, 'P', channel_3_phase_, '*');
send_configuration(1, 'D', channel_1_duty_, '*');
send_configuration(2, 'D', channel_2_duty_, '*');
send_configuration(3, 'D', channel_3_duty_, '>');
}

void loop() {
     if (Serial.available()) {int a = Serial.read();_mySerial.write(a);Serial.write(a);}
}

void send_configuration (uint8_t _ch, uint8_t _conf, uint16_t* _value, uint8_t _end)
{char _buffer[12]; // Buffer
// Prepare the buffer Note: \n\r is not necessary
sprintf(_buffer, "%d%c%d.%02d%c\n\r",_ch,_conf,_value[0],_value[1],_end);
while(!digitalRead(Rx_Ready)); delayMicroseconds(50); // Wait for the Pwm-Core to be ready
_mySerial.write(_buffer); // Send buffer to Pwm-Core
Serial.write(_buffer);
}

//End of Code
```

Outputs:
1U500.05<
2M10.00*
3M15.38*
1P0.00*
2P120.50*
3P240.00*
1D25.00*
2D25.17*
3D50.00>

PRODUCT CODE



TECHNICAL DRAWING



CONTACT INFORMATION

Lentark Electronics

Website : www.lentark.com
E-mail : info@lentark.com