PAK-VIIIa Pulse Coprocessor Data Sheet

AWC
310 Ivy Glen
League City, TX 77573
(281) 334-4341
http://www.al-williams.com/awce.htm

V1.6 30 Aug 2003

Table of Contents

Overview

The PAK-VIII is an 8 channel pulse output coprocessor. It can perform a variety of functions:

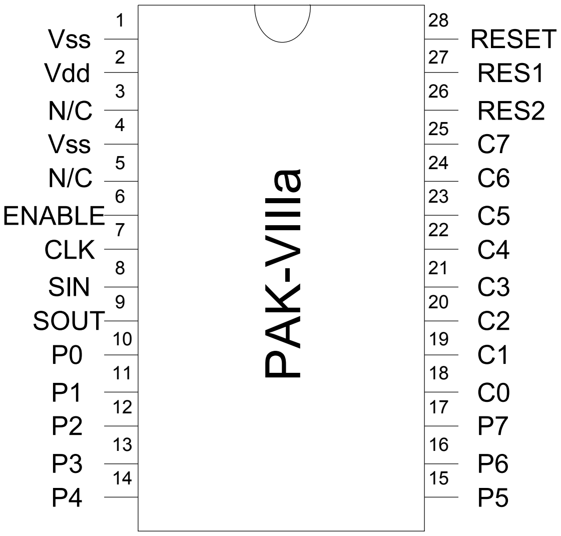
- Generate pulses continuously
- Generate a certain number of pulses and then stop
- Create pulses with varying duty cycles
- Count generated rising edges or falling edges
- Control servos
- Generate PWM
- Works with Basic Stamp's SHIFTIN and SHIFTOUT
- Easy to use

Like all PAKs, the PAK-VIII is simple to connect to a Stamp or any microcontroller. If your microcontroller can switch pins between input and output status, you can connect a single PAK with as few as 2 I/O lines. However, the PAK will allow you to use separate I/O pins (3 lines) if necessary. You can also connect multiple PAKs together using the same two or three lines if you provide an additional enable line for each PAK.

The PAK-VIII is a standard 28-pin IC. In order to operate, it must have a regulated supply of 5V and connection to a clock element. The PAK-VIII includes a 50MHz ceramic resonator that you can use to clock the chip. If you need more accuracy, a crystal or external oscillator may be used.

If You Need Help

If you require assistance with your PAK VIIIa, please feel free to contact us. For best support, e-mail stamp@al-williams.com. However, you may also call between 9AM - 4PM Central Time at (281) 334-4341. You can also fax to (281) 754-4462. Be sure to check out our Web page for updates at www.al-williams.com/awce.htm.



WARNING: The PAK VIII is a static-sensitive, CMOS device. Observe static precautions when handling. Operating the device without both Vss pins grounded or with RES1 or RES2 disconnected may damage the chip.

Pin Connections

Pin	Name	Type	Description
1	Vss	Power	Ground
2	Vdd	Power	+5V
3, 5	N/C	N/C	No connection
4	Vss	Power	Ground
6	Enable	Input	If this pin is not connected or high, the PAK is active. Otherwise, the PAK does not drive the SOUT line, nor does it respond to commands.
7	CLK	Input	Used to clock data to and from the PAK.
8	SIN	Input	Data Input
9	SOUT	Open Collector Output	Data Output
10 to 17	P0-P7	Outputs	Pulse output channels
18 to 25	C0-C7	Outputs	Clock output channels
26	RES1	Clock	Connects to resonator (75MHz max)
27	RES2	Clock	Connects to resonator
28	RESET	Input	Hardware resets the PAK when low. Must be high for normal operation.

Command Format

The PAK-VIII uses a very simple command byte. Each channel (P0 to P7) has 6 associated registers. To write a register you form a single byte with the following bits:

7	6	5	4	3	2	1	0
0	0	R2	R1	R0	C2	C1	C0

- R2-R0 is the register number from the table below.
- C2-C0 is the channel number in binary (0=000, 1=001, 2=010, 3=011, 4=100, 5=101, 6=110, 7=111).

Here are the available registers:

Register number	Name	Function	Units (prescale normal)
000 (0)	DUR	Current pulse duration	10uS
001 (1)	DURHIGH	Duration of high part of output	10uS
010 (2)	DURLOW	Duration of low part of output	10uS
101 (5)	COUNT	Number of pulses to generate	
110 (6)	CTRL	Sets modes (count and enable)	

Each time you send a register write command (bit 7 and 6 both = 0) to the PAK, it expects two bytes. The first byte is the least significant byte of a 16-bit word. The second byte is the most significant byte. Bytes are sent most significant bit first. You can find more details about the PAK protocol (including example code) at http://www.al-williams.com/doclib.htm.

In addition to the write command, you can use the following special commands:

Command	Description
11111111	Reset everything
1100PPPP	Set prescaler (affects all timing functions)
	PPPP = 0000 = 1:2 (default; 10uS)
	PPPP = 0001 = 1:4 (20uS)
	PPPP = 0010 = 1:8 (40uS)
	PPPP = 0011 = 1:16 (80uS)
	PPPP = 0100 = 1:32 (160uS)
	PPPP = 0101 = 1:64 (320uS)
	PPPP = 0110 = 1:128 (640uS)
	PPPP = 0111 = 1:256 (1280uS)
	PPPP = 1000 = 1:1 (not useful when using more than 4 channels; 5uS)

010XXCCC	Sets status of channel CCC:
	XX = 00 = Force low
	XX = 01 = Force high
	XX = 10 = Hi impedance state (default)
	XX = 11 = Pulse output operation

Command	Description
10RRRCCC	Read register RRR (two bytes sent back to host; least significant first).
11111110	Disable all output and reset all counters
11111101	Enable outputs (default state; disabling the PAK, setting all channels, and then enabling allows you to sync the leading edges of the pulse outputs, if required)

Resetting the PAK

There are three ways you might reset the PAK-VIII. When you start, your host microcontroller might take a short time to take control of the PAK's I/O lines. That's why it is important to initially reset the PAK's communications system.

Resetting the communications system is easy. Simply pull the SIN line low and raise the CLK input. Then, while CLK is high, raise SIN. When you release the clock, the PAK will reset its communications. This does not affect any running counters or timers. It also does not reset any options (for example, pull up resistors). You can perform this reset using the FReset subroutine included with the example Basic Stamp library.

If you want to force a hard reset, which will clear all of the aforementioned items, send the PAK a $FF or use the library's FTotalReset command.

Another way to force a hard reset is to pull the RESET pin low briefly and then restore it to high. This will physically reset the PAK-VIII, destroying any settings, counts, or intervals in progress. Notice that most of the time this is not necessary so there is no need to connect the RESET pin to the host microcontroller. Simply connect it to +5V.

Registers

Each channel has several identical 16-bit registers. These register control the PAK's operation. Initially, all channels are disabled and set to high impedance state. This allows your program time to set the registers before enabling the outputs.

Register 1 sets the duration of the high-going part of the pulse. Register 2 sets the duration of the low-going part of the pulse. With the default prescaler setting, these registers operate in 10uS intervals. However, you can alter the prescaler to change this.

Register 0 tracks the current amount of time for the current half cycle -- you'll rarely need to access this register.

Register 6 is the control register. This register really only uses 3 bits. The bottom bit (bit 0) determines if the channel is enabled (a 0 means that it is enabled). Bit 1 is set if the PAK is to output a certain number of pulses and then stop. Finally, bit 7 shows the current state of the output pin (read-only).

If counting is enabled (bit 1 in register 6 is set) then the PAK examines register 5. The PAK treats the register as a flag in bit 15 followed by a 15-bit counter. If the flag is clear (that is to say the number is <$8000) then the PAK generates the requested number of high pulses when the channel is enabled. Register 5 counts down to 0 and at the end, the output will be low.

If register 5 has bit 15 set (that is the number is >$8000), the PAK counts the low pulses. In this case, the counter counts down to $8000 and at the end the output will be high. You can read the counter register (register 5) to determine if the PAK has completed the pulse train or is still generating pulses.

You can access these registers by forming the appropriate request byte and sending it to the PAK. The example Basic Stamp library has a function, FCommand, that makes this easier. To use FCommand, set Register to the register number you want and Chan to the channel number. The value you wish to store is in the fpx variable.

The routines do not change the input variables, so if you set register to 0, for example, you don't have to set it again until you want to change its value no matter how many calls to FCommand you make.

Special Commands

In addition to reading registers, you may execute special commands. Sending all 1's to the PAK, as mentioned before, causes a hard reset. This will terminate and reset all pulse functions. It also resets all options including prescaling.

If you need to generate pulses longer than the maximum allowed, you can reduce the speed of the PAK using a slower ceramic resonator or crystal. However, an easier solution is to engage the prescaler. Using the prescaler, you can change the time base from 5uS to 10uS to 1280uS in 8 steps. This changes the time for all channels.

The default setting for the prescaler to 1:2. This means that all pulse widths are normally in units of 10uS. You can reduce to 1:1 for 5uS resolution. However, when doing so, the PAK can only service 4 channels at once. You can enable any 4 channels, but enabling more than 4 with a 1:1 prescale will cause inaccuracy in the generated pulses.

The prescaler also determines how long a pulse you can generate. Since each register is 16-bit, at 10uS, the largest high (or low) you can generate is just over 650mS.

You can set the prescaler using the Basic Stamp library's FPrescale function (set fpb to the correct prescale code you want to use). The prescale command does not return any data to the host.

The other special commands allow you to force a channel to a certain state, or read a register. You can force a channel to be a high impedance pin, a logic 0, or a logic 1. Setting any of these states automatically disables the channel in question. Setting the normal state automatically enables the channel.

By default, pins C0-C7 are weakly pulled up to 5V. However, you can write to the low byte of register 7 channel 1 to cause a repeating clock to appear on these pins. A 0 in the corresponding bit positions will enable the clock output bits. C0 will cycle at the sample rate (10uS by default), C1 will cycle at double that time (20uS), C2 doubles C1 (40us), etc.

Software

If you are using the Basic Stamp II or IISX, you can use the simple library included on the companion disk to work with the PAK-VIII. You'll need to change the DATAP, DATAPIN, and CLK variables to match your circuit.

Here are the available subroutine calls:

Call	Arguments	Description

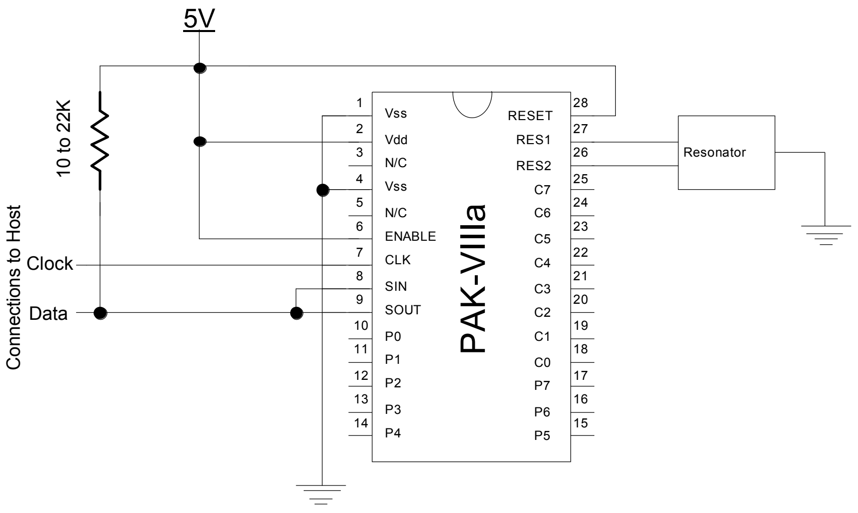
FCommand	chan = channel number
	
	register = register number
	
	fpx = value
		Set register

FReset	none	Reset PAK I/O
FTotalReset	none	Completely reset PAK
FPrescale	fpb=PPPP	Set prescale ratio

FreadReg	chan = channel number
	
	register = register number
		Returns value in fpx

FJam	chan = channel number
	
	fpb = two bit code
		Set state (00 = low; 01 = high; 10 = hi-z; 11 = normal)

Example Circuit



Typical Circuit

Using the PAK-VIII for PWM

The PAK-VIII can easily generate pulses appropriate for Pulse Width Modulation (PWM). The ratio of the high and low durations will determine the PWM output. For example, if you turn a pulse on for 10uS and off for 30uS that will be a duty cycle of 25% (10 divded by 10+30). If you turn the pulse on for 20uS and off for 20uS that is a 50% duty cycle.

PWM is useful for generating voltages (with an RC integrator) and for controlling heating elements, motors, and similar devices.

Using the PAK-VIII for Servo Control

The PAK-VIII can also easily control servo motors. Servos operate by altering their position based on a pulse width. Usually a 1.5mS pulse will center the servo. Shorter pulses cause the servo to move in one direction. Longer pulses reverse the direction. The servo's final position depends on the pulse width.

It is simple to center a servo by turning a pulse "on" for 150 units (assuming the default 10uS prescale is in force). You can turn the pulse "off" for a longer time, perhaps 20mS (a count of 2000). Then by adjusting the "on" time, you can control the servo position easily. Once set, the PAK-VIII will continue to supply the control pulses without further intervention from the host computer.

This can be especially useful when using servos modified for continuous rotation. Combined with the counting function of the PAK-VIII you can deliver a preset number of pulses to the servo with a single command sequence from the host.

Frequently Asked Questions

Q: Can I use a different clock?

A: Yes, you can use a different ceramic resonator or crystal to change the speed of the PAK-VIII. Reducing the speed will reduce the power required, and also stretch the pulse duration. A 25MHz resonator, for example, will generate pulses in 20uS intervals (by default; the prescaler may adjust this). The maximum frequency of operation is 75MHz.

Q: I'm writing my own ShiftIn and ShiftOut instructions to communicate with the PAK. How fast can I go?

A: The Stamp II's ShiftIn and ShiftOut instructions operate at about 16kHz. However, with a 50MHz resonator, the PAK should operate fine when the CLK line operates at 100kHz.

Q: How accurate are the pulse widths generated?

A: The pulse widths are as accurate as the timing element connected. For the ceramic resonator supplied, that is reasonably accurate. However, you can replace the reasonator with a 50MHz crystal (and appropriate capacitors) to improve the accuracy if you wish. Remember too that the time it takes you to communicate with the PAK is a factor as well. Using a Stamp II, reading or writing 16 bits from the PAK takes about 1mS.

Q: How do I connect the resonator?

A: The middle pin is ground. The outer pins are interchangeable.

Notes

Notes

Specifications

Absolute Maximum Ratings

Ambient temperature under bias	-40°C to +85°C
Storage temperature	-65°C to +150°C
Voltage on VDD with respect to VSS	0 to +7.0V
Maximum current out of VSS pin	130 mA
Maximum current into VDD pin	130 mA

DC Characteristics

Parameter	Minimum	Typical	Maximum
Supply voltage	3V	5V	5.5V
Vdd rise time on power up	.05V/ms	-	-
Supply current @ 5V/50MHz	-	77mA	