AWCE Home
Math Coprocessors
I/O Coprocessors
PWM Coprocessor
Keyboard Interface
Pulse Coprocessor
Stamp Book
Stamp Prototyping
Lab Kit
Stamp II Carrier
PIC Programming
Full Order Form
Project of the Month
Free Emulator
Updates

Basic Stamp Circuit of the Month

Here's where I'll try to show you something we've done with Basic Stamps month. Always quick and dirty, and the month is plus or minus a few days These circuits are easy to try with our economical ASP-II and even easier companion lab kit.

March 2000

Timing Pulse

The latest entry in our PAK coprocessors is the PAK-VII, a pulse input cop You can read the datasheet, but here is the nickel tour:

- 8 channels of pulse measurement
- Measures high and low durations with 5uS resolution
- Counts rising and falling edges
- Software select CMOS/TTL, Pullups, and Schmitt trigger inputs
- Access to raw timing and input state
- Provides 200uS and 1 second timer ticks
- Uses the PAK two-wire protocol (similar to SPI) -- easy to program, spe independent, easy to use without interrupts

With the Stamp you can use ShiftIn and ShiftOut to communicate with the use our simple Stamp library). Even while the Stamp is doing something e PAK-VII is measuring and counting. (Yes, there is a Pulse Output coproce works.)

A Clock

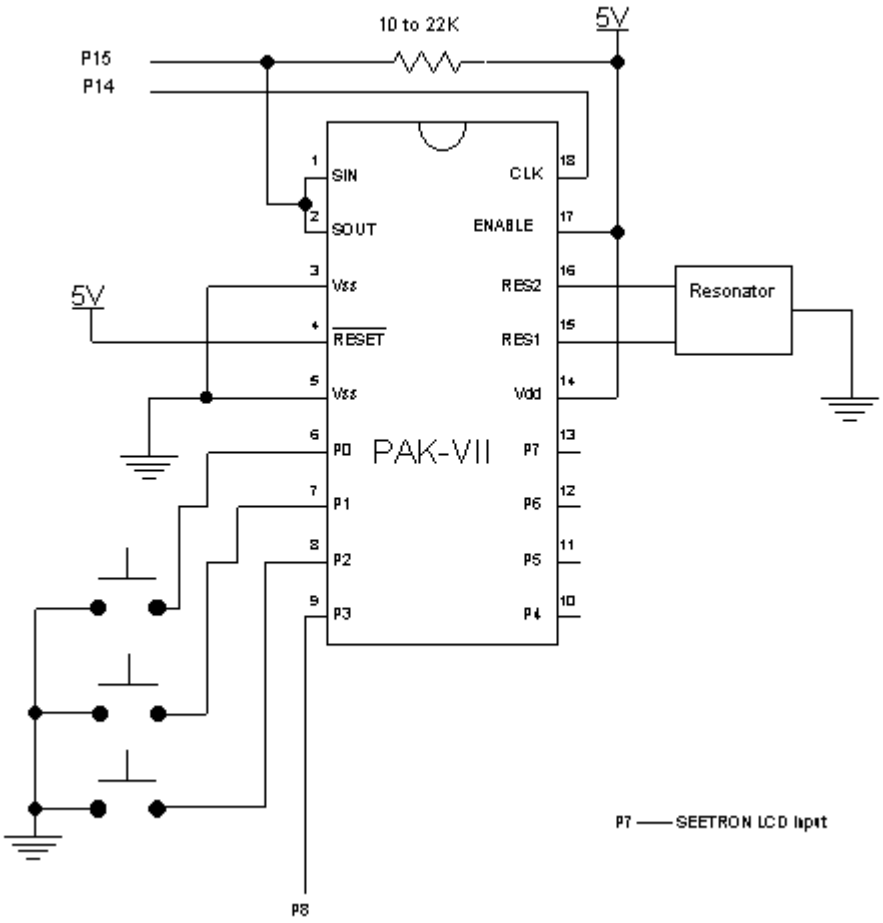
This month's circuit is a 24 hour clock using a Stamp II, a PAK-VII, and a serial LCD. The Stamp is free to do other operations without losing time. I case, it scans push buttons and also measures a frequency. The program generates a test frequency for you to measure, or you can hook it up to ar signal if you like.

The circuit has three push buttons connected to the PAK-VII. Honestly, th is better at measuring pulses than push buttons. However, it can do the jo leaves Stamp pins free. If the buttons were bounceless, the count feature allow the PAK-VII to handle multiple key presses. However, since there is bounce on the switches and I didn't want to add hardware to debounce th handle the debounce in the Stamp software.

One button enters Set mode. The LCD will show an asterisk by the time w

Set mode is active. While in set mode, the two other buttons will incremen
and minute respectively. The Stamp only checks the buttons every second
easy to see that the PAK-VII is monitoring the buttons even when the Star

When you depress the Set button again, the clock exits Set mode and res
second count to 0 seconds.



Under the Hood

The PAK-VII uses a register structure.

Here are the available registers:

Register number	Name	Function	Units
000	DURLOW	Duration of last logic low pulse	5uS
001	DURHIGH	Duration of last logic high pulse	5uS
010	RAWLOW	Current count for low pulse	5uS
011	RAWHIGH	Current count for high pulse	5uS
100	RISE	Number of rising edges detected	
101	FALL	Number of falling edges detected	
110		Special registers	

110		Special registers	
	TICK200	Channel 0 - Timer tick	200mS
	TICK	Channel 1 - Timer tick	1S
	INP	Channel 7 - Current input status	
111	SUM	Sum of registers 000 and 001	5uS

When using the Basic Stamp library, you set the channel number (0-7), th number (0-7) and you can set a flag that will clear the channel after you re you can clear the entire set of registers for the channel. Then you call FC and read the result from FPX. Here are the commands in the library.

Call	Arguments	Description
FCommand	clearreg = 1 to clear register clearchan = 1 to clear channel chan = channel number register = register number	Read register to fpx (do n set clearreg and clearchan simultaneously)
FReset	none	Reset PAK I/O
FTotalReset	none	Completely reset PAK
FPrescale	fpb=PPPP	Set prescale ratio
FPullUp	fpx=pull up byte	Set pull up byte (0=pullup
FThresh	fpx=threshold byte	Set threshold (0=CMOS)
FSchmitt	fpx=Schmitt byte	Set Schmitt trigger (0=on)

The Code

```
' Clock and frequency counter
' using BS2 and PAK-VII -- Williams

' Change these to suit your setup
datap con 15   ' Data pin (I/O)
datapin var in15
clk con 14     ' Clk pin (output)
' SEETRON LCD and baud rate
lcd con 7
baud con 84+$4000

' Register names
DURLOW con %000
DURHIGH con %001
RAWLOW con %010
RAWHIGH con %011
RISE con %100
FALL con %101
TICK200 con %110
TICK con %110
INP con %110
SUM con %111


output clk
output datap

fpx var word        ' Integer used by some routines
```

```
fpb var byte        ' byte

' parameters for FCommand
chan var nib
register var nib
clearreg var bit
clearchan var bit

setmode var bit  ' true when setting time
seconds var byte
last var byte
minu var byte
hour var byte

gosub freset        ' always reset!

' Main code
gosub FTotalReset  ' reset PAK-VII hard
serout lcd,baud,[12,14]  ' clear display and backlight on
fpx=0
' Turn on Pullup and Schmitt triggers
gosub FPullUp
gosub FSchmitt

' read & clear timer (chan 1, TICK)
tloop:
clearreg=1
chan=1
register=TICK
gosub FCommand

' get correct time units
seconds = seconds + fpx
minu = minu + (seconds/60)
seconds = seconds // 60
hour = hour + (minu/60)
minu = minu // 60
if hour<>24 then nomidnite
hour=0  ' wrap around at 23:59 + 1
nomidnite:

' update display etc no more than once per second
if last=seconds then tloop
last=seconds

' out to LCD
serout lcd,baud,[12,dec2 hour,":",dec2 minu,":",dec2 seconds]
' only print * if in set mode
if setmode=0 then nodset
serout lcd,baud,[" *"]

' see if any button pushes for hour set
chan=1
register=FALL
gosub FCommand
if fpx=0 then noseth
hour=hour+1//24
pause 1
gosub FCommand  ' clear bounces after 1ms

noseth:
' see if any button pushes for minute set
chan=2
gosub FCommand
```

```
if fpx=0 then nodset
minu=minu+1//60
pause 1
gosub FCommand  ' clear bounces after 1ms

nodset:   ' no setting

' generate test frequency
' (about 806Hz on a BS2)
for fpx=1 to 10
toggle 8
fpx=fpx
toggle 8
next

' frequency meter
chan=3
register=SUM
gosub FCommand

' Convert to Hz 1000000/fpx is about the same
' as 50000/(fpx/4)
fpx=fpx/4
fpx=50000/fpx  ' fpx= approx freq in hZ
serout lcd,baud,[13,"f=",dec fpx," Hz"]

' check for set mode change
chan=0
register=FALL
gosub FCommand
if fpx=0 then tloop
if setmode then noset
' start setting time
  setmode=1
goto tloop

noset:
' stop setting time
  seconds=0
  last=59
  setmode=0
  goto tloop

' Reset the Pak7 I/O
FReset:
LOW DATAP
LOW CLK
HIGH CLK
HIGH DATAP
LOW CLK
return

FCommand:
' inputs chan = channel, register = register, clearreg and clearch
 fpb=(clearchan<<7) + (clearreg<<6) + (register<<3) + chan
 gosub FSendByte
FReadWord:
  Shiftin datap,clk,MSBPRE,[fpx.lowbyte,fpx.highbyte]
  return

FTotalReset:
  fpb=$FF
FSendByte:
  Shiftout datap,clk,MSBFIRST,[fpb]
```

```
  return

FReadByte:
  shiftin datap,clk,MSBPRE,[fpb]
  return

FPrescale:
' set fpb to the prescale constant
  fpb = fpb + $C0
  goto FSendByte

FPullUp:
' set fpx to the pull up mask
  fpb=%11010000
Fscmd:
  gosub FSendByte
  fpb=fpx
  goto FSendByte

FThresh:
' set fpx to the threshold mask
  fpb=%11010001
  goto Fscmd

FSchmitt:
' set fpx to the Schmitt trigger mask
  fpb=%11010010
  goto Fscmd
```

FeedBack

Are you reading these projects regularly? Are they helpful? What would yo
see? How about PIC projects some months instead of Stamps? Do you pr
Stamp I or Stamp II projects? Take a moment to share your thoughts on th
Project of the Month.

Back Home
