Summary

The following software shows how to set up and read the pulse accumulator.

```
// Copyright 1997 Mosaic Industries, Inc.  All Rights Reserved.
//  Disclaimer: THIS SOFTWARE IS PROVIDED ON AN "AS IS" BASIS, WITHOUT ANY
//  WARRANTIES OR REPRESENTATIONS EXPRESS OR IMPLIED, INCLUDING, BUT NOT
//  LIMITED TO, ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS
//  FOR A PARTICULAR PURPOSE.

// This program demonstrates how to set up and read the pulse accumulator
// which counts pulses on PortA bit 7.
// Simply connect your pulse source (0 to 5 volts) to input PA7
// (pin 3 on the QED Digital I/O connector).
// See the "Programmable timer" chapter in the pink "HC11F1" book,
// and the description in the "Pulse Accumulator" section of the
// "Programmable Timer and Pulse Accumulator" chapter in the QED Hardware
// Manual.

// We take as a simple example an application that requires some
// action to be taken after every TARGET_NUM_PULSES have occurred.


#include <\mosaic\allqed.h>
 // this include statement should appear at the top of each source code file.

// the following registers are defined in \fabius\include\mosaic\qedregs.h:
// TMSK2 and TFLG2: timer interrupt masks and flags
// PACTL: pulse accumulator control register
// PACNT: pulse/time count register
// DDRA:  porta direction register
// masks for PACTL register:
#define  PAEN  0x40      // pulse accumulator/gated enable bit
#define  PAMOD 0x20      // mode select bit, clear selects pulse accum
#define  PEDGE 0x10      // signal edge trigger/gate configuration bit

// masks for TFLG2 register:
#define  PAOVF 0x20      // pulse accumulator/gated enable bit
#define  PAIF  0x10      // trigger/gate sense flag bit

// masks for TMSK2 register:
#define  PAOVI 0x20      // interrupt enable bit for PACNT overflows
#define  PAII  0x10      // interrupt enable bit for edge detection

// masks for DDRA register:
#define  PA7_MASK 0x80      // set: output; clear: input
```

```
_Q void RisingEdgePulseAccum(void)
// initializes pulse accumulator to sense rising edges.
{   PACTL |= PEDGE;     // rising edge
    PACTL &= !PAMOD;    // set pulse accumulation mode
    PACNT = 0;          // init the count
}

_Q void FallingEdgePulseAccum(void)
// initializes pulse accumulator to sense falling edges.
{   PACTL &= !PEDGE;    // falling edge
    PACTL &= !PAMOD;    // set pulse accumulation mode
    PACNT = 0;          // init the count
}

_Q void EnablePulseAccum(void)
// allows counting to start; result is in PACNT register.
// execute AFTER RisingEdgePulseAccum() or FallingEdgePulseAccum().
{   DDRA &= !PA7_MASK;          // make pa7 an input
    TFLG2 &= !(PAOVF | PAIF);  // clear both interrupt flags
    PACTL |= PAEN;              // enable the pulse accumulator
}

_Q void DisablePulseAccum(void)
// stops the counting
{   PACTL &= !PAEN;
}

// ********************* SIMPLE EXAMPLE APPLICATION ******************

#define  TARGET_NUM_PULSES 10      // we'll take action after every 10 pulses
  // NOTE: target must be less than 127 in this example because
  // we use 8-bit signed math.
  // A more powerful application is easy to construct by
  // incrementing a variable every time the PACNT register overflows
  // from 0xFF to 0x00.

static char prior_num_pulses;      // holds latest 8-bit accumulator count
static int  action_flag;      // incremented each time Action() is called

_Q void Action(void)
// default action; called by CheckPulses()
{   action_flag += 1;
}


_Q void CheckPulses(int target)
// takes action if at least target pulses have elapsed since last action
// call this from your application program's main loop
{   if((PACNT - prior_num_pulses) >= target)
        {   prior_num_pulses = PACNT;  // update the variable
            Action();                  // call the action routine
        };
//  else Pause();     // optional, for multitasking applications
}
```

```
_Q void InitSystem(void)
{   prior_num_pulses = 0;     // note: init all variables here!!
    action_flag = 0;
    FallingEdgePulseAccum();
    EnablePulseAccum();
}



void main( void )
// this infinite loop beeps after every 10 pulses
{ InitSystem();
  while(1)
      CheckPulses(TARGET_NUM_PULSES);
}
```


Mosaic Industries

5437 Central Ave Suite 1, Newark, CA 94560 Telephone: (510) 790-8222 Fax: (510) 790-0925