MSP430 Launchpad Shift Register

From Texas Instruments Wiki

by Andrew Morton

This is a quick example of using an 8-bit shift register to drive eight output pins with only three outputs of the MSP430. By chaining the serial out to serial in, and tieing the clock and latch pins, one can add even more shift registers, and more outputs, while still using just three pins.

Contents

Quick Start

1. Get external components.
   - *1 Resistor* of **1KOhm**
   - *8 Resistors of* **270Ohm**
   - *1 Capacitor of* **.1uF** (may optionally be added between the latch pin and ground to stabilize the signal)
   - *8 LED's:* **Just about any normal LED will do**
   - *1 IC*: **74HC595 Shift '***Register' or equivalent chip*
   - *Small breadboard and jumper cables*
2. Create new project in CCS/IAR and copy/paste code
3. Build circuit.

Development

This example drives eight LED's in a "ping-pong" pattern using the 74HC595 shift register, on P1.0, P1.4, P1.5, and optionally, P1.6. If you do opt not to control the OE pin(13), then tie it directly to GND

Schematic

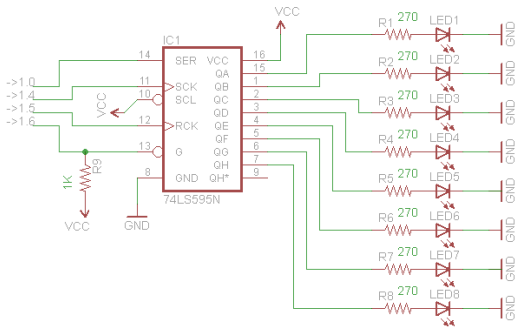

Fig. 1: Schematic for Shift Register Circuit

Link to video showing the breadboarded and running circuit: www.youtube.com/watch (http://www.youtube.com/watch?v=8IT7qNJXEvc)

Code

```
//******************************************************************************************
// MSP430 Driver for 74HC595 Shift Register
//
// Description; Drives 8 LED's with 3 digital pins of the MSP430, via a shift register
//
// MSP430x2xx
//
//******************************************************************************************
#include <msp430x20x2.h>

//Define our pins
#define DATA BIT0 // DS -> 1.0
#define CLOCK BIT4 // SH_CP -> 1.4
#define LATCH BIT5 // ST_CP -> 1.5
#define ENABLE BIT6 // OE -> 1.6
// The OE pin can be tied directly to ground, but controlling
// it from the MCU lets you turn off the entire array without
// zeroing the register

// Declare functions
void delay ( unsigned int );
void pulseClock ( void );
void shiftOut ( unsigned char );
void enable ( void );
void disable ( void );
void init ( void );
void pinWrite ( unsigned int, unsigned char );


int main( void )
{
  // Stop watchdog timer to prevent time out reset
  WDTCTL = WDTPW + WDTHOLD;
  P1DIR |= (DATA + CLOCK + LATCH + ENABLE);  // Setup pins as outputs
  enable(); // Enable output (pull OE low)

  int i;
  //Do a "ping-pong" effect back and forth
  for(;;){
    for ( i = 0 ; i < 8 ; i++ ){
      shiftOut(1 << i);
      delay(50);
    }
    for ( i = 7 ; i >= 0 ; i-- ){
```

```
      shiftOut(1 << i);
      delay(50);
    }
  }
}

// Delays by the specified Milliseconds
// thanks to:
// http://www.threadabort.com/archive/2010/09/05/msp430-delay-function-like-the-arduino.aspx
void delay(unsigned int ms)
{
 while (ms--)
    {
        __delay_cycles(1000); // set for 16Mhz change it to 1000 for 1 Mhz
    }
}

// Writes a value to the specified bitmask/pin. Use built in defines
// when calling this, as the shiftOut() function does.
// All nonzero values are treated as "high" and zero is "low"
void pinWrite( unsigned int bit, unsigned char val )
{
  if (val){
    P1OUT |= bit;
  } else {
    P1OUT &= ~bit;
  }
}

// Pulse the clock pin
void pulseClock( void )
{
  P1OUT |= CLOCK;
  P1OUT ^= CLOCK;

}

// Take the given 8-bit value and shift it out, LSB to MSB
void shiftOut(unsigned char val)
{
  //Set latch to low (should be already)
  P1OUT &= ~LATCH;

  char i;

  // Iterate over each bit, set data pin, and pulse the clock to send it
  // to the shift register
  for (i = 0; i < 8; i++)  {
      pinWrite(DATA, (val & (1 << i)));
      pulseClock();
  }

  // Pulse the latch pin to write the values into the storage register
  P1OUT |= LATCH;
  P1OUT &= ~LATCH;
}

// These functions are just a shortcut to turn on and off the array of
// LED's when you have the enable pin tied to the MCU. Entirely optional.
void enable( void )
{
  P1OUT &= ~ENABLE;
}

void disable( void )
{
  P1OUT |= ENABLE;
}
```

Future Ideas

- Add support for chained shift registers (see Talk Page)
- Take different/larger datatypes for larger arrays

References

1. MSP430x2xx Family User Guide (http://www.ti.com/litv/pdf/slau144e)
2. MSP430G2x31 Datasheet (http://www.ti.com/lit/gpn/msp430g2231)

Links

Amplifiers & Linear (http://www.ti.com/lsds/ti/analog/amplifier_and_linear.page)
Audio (http://www.ti.com/lsds/ti/analog/audio/audio_overview.page)
Broadband RF/IF & Digital Radio (http://www.ti.com/lsds/ti/analog/rfif.page)
Clocks & Timers (http://www.ti.com/lsds/ti/analog/clocksandtimers/clocks_and_timers.page)
Data Converters (http://www.ti.com/lsds/ti/analog/dataconverters/data_converter.page)
DLP & MEMS (http://www.ti.com/lsds/ti/analog/mems/mems.page)
High-Reliability (http://www.ti.com/lsds/ti/analog/high_reliability.page)
Interface (http://www.ti.com/lsds/ti/analog/interface/interface.page)
Logic (http://www.ti.com/lsds/ti/logic/home_overview.page)
Power Management (http://www.ti.com/lsds/ti/analog/powermanagement/power_portal.page)
Processors (http://www.ti.com/lsds/ti/dsp/embedded_processor.page)

- ARM Processors (http://www.ti.com/lsds/ti/dsp/arm.page)
- Digital Signal Processors (DSP) (http://www.ti.com/lsds/ti/dsp/home.page)
- Microcontrollers (MCU) (http://www.ti.com/lsds/ti/microcontroller/home.page)
- OMAP Applications Processors (http://www.ti.com/lsds/ti/omap-applications-processors/the-omap-experience.page)

Switches & Multiplexers (http://www.ti.com/lsds/ti/analog/switches_and_m
Temperature Sensors & Control ICs (http://www.ti.com/lsds/ti/analog/temperature_sen
Wireless Connectivity (http://focus.ti.com/wireless/docs/wirelessoverview familyId=2003&sectionId=646&tabId=2735)