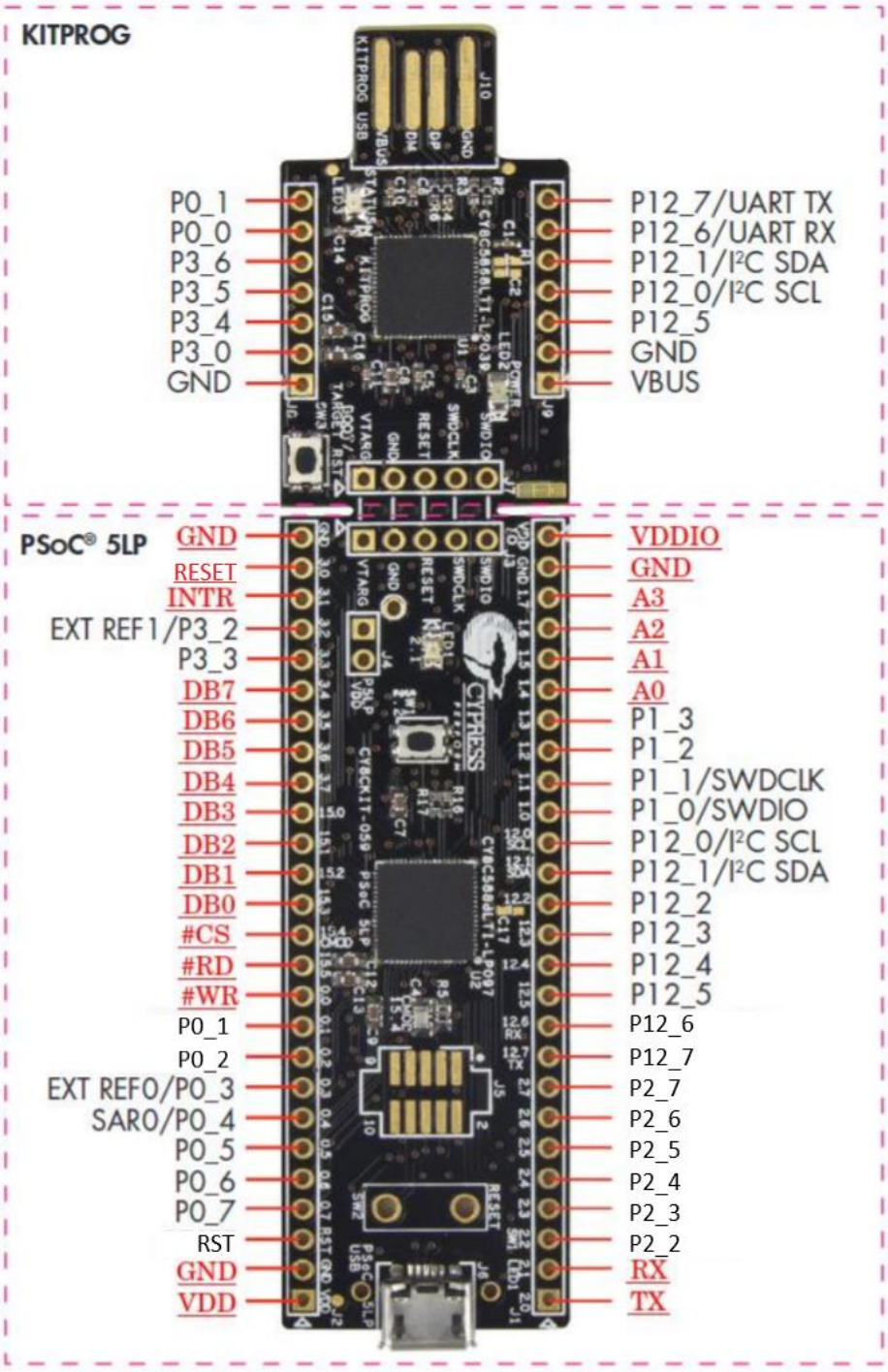
How to use the PSoC – based 16C450 Replacement

Matthew Burns
Eric Ponce

August 2017 (Updated April 2018)



1 Overview

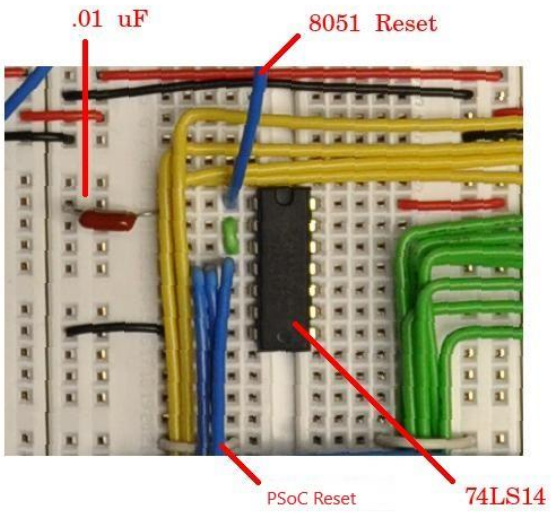
The PSoC based 16C450 Replacement is intended to replace the 16C450 serial communication chip. It has a data bus, an address bus, and all other supporting pins to make this PSoC communicate with an 8051 as if it were an 8000-series peripheral with 16 registers. In addition to that, it has TX and RX pins to carry out UART communication with other UART peripherals. Furthermore, the firmware serves as a starting point to create different combinations of peripherals for the 8051.

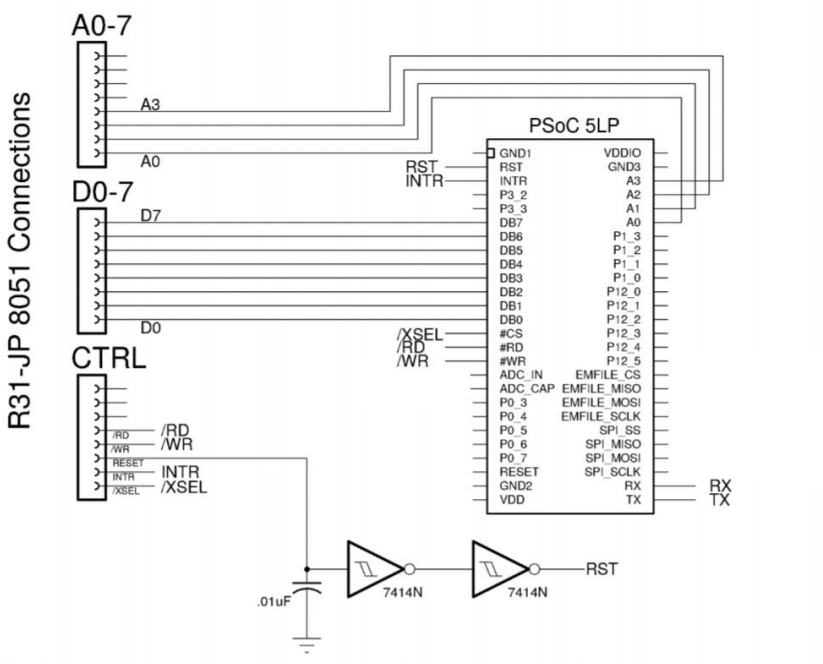
2 Hardware

This 16C450 replacement is designed around a CY8CKIT-059 PSoC 5LP. Many of the pins on this PSoC are not used in this implementation and are labeled in black on the pinout shown on page one of this manual. The pins that are used are labeled with their function in red.

This PSoC is fitted with an 8-bit data bus, a 4-bit address bus, /RD and /WR lines, a /CS line, and an INTR line to communicate with the 8051 as if it were an 8000-series peripheral. It also has a RESET line which controls the software reset of the entire PSoC replacement. This RESET line should be connected to the 8051's reset line through two 7414 inverters – with a small (~.01 uF) capacitance on the 8051's reset line for filtering.

Note: The RESET pin is labeled 15.3 on the PSoC and is different then the pin labelled 'RST' on the development board, which is not used in this application.





*Note: Power (5V) and ground (0V) connections were intentionally left off of the PSoC in the schematic for clarity.

In addition to connecting all standard lines for 8000 series communication, the three GND pins of the PSoC should connect to the same ground as the 8051. VDD and VDDIO should be tied to +5VDC. RX is the receive line for the PSoC UART component and is intended to be tied to the transmit line of the device you wish to communicate with. TX is the transmit line for the PSoC UART component and is intended to be tied to the receive line of the device.

3 Firmware

The firmware for this 16C450 replacement was created in PSoC Creator 3.3.

Interface:

The UART module makes use of the first three of the 16 addressable registers on the PSoC: address 0x00, address 0x01, and address 0x02.

Address 0x00 is the control register for the UART module. It allows the user to turn on and off the UART module, set the baud rate of the UART module, and manipulate the receive / transmit flags and the receive / transmit interrupt enable bits.

Address 0x01 is the UART write register. Writing to this register causes the UART module to transmit the byte that was written to this register provided that the UART module is on and is not currently transmitting a byte.

Address 0x02 is the UART read register. Reading this register returns the byte that was most recently received by the UART module. Reading this register does NOT clear the receive flag in the control register

All these registers have a reset value of 0x00.

UART Registers

Address:	Register Name:	Reset Value:	Function:
0x00	UART Control Register	0x00	The control register allows you to turn on and off the UART module, set the baud rate, and control the UART interrupts.
0x01	UART Transmit Buffer	0x00	The transmit buffer is the register you write to when you wish to transmit a byte.
0x02	UART Receive Buffer	0x00	The receive buffer holds the value of the most recently received byte.

Control Register:

7	6	5	4	3	2	1	0
TF	RF	TIE	RIE	M3	M2	M1	M0

The bits of the control register (0x00) are set up as follows:

Bit 7: Transmit Flag. This bit is automatically set when the UART component finishes transmitting a byte. Both this bit and the receive flag (bit 6) need to be cleared to reset the interrupt line if this UART component sent a falling edge interrupt to the 8051.

Bit 6: Receive Flag. This bit is automatically set when the UART component receives a byte. Both this bit and the transmit flag (bit 7) need to be cleared to reset the interrupt line if this UART component sent a falling edge interrupt to the 8051.

Bit 5: Transmit Interrupt Enable. If this bit is set, the PSoC will send a falling edge interrupt to the 8051 upon completion of a UART transmit.

Bit 4: Receive Interrupt Enable. If this bit is set, the PSoC will send a falling edge interrupt to the 8051 upon receiving a byte.

Bit 3: (M3) The lower nibble of this control register is used to turn on and off the UART module and set the baud rate.

Bit 2: (M2)

Bit 1: (M1)

Bit 0: (M0)

The UART mode is used to control the baud rate of the UART module and is determined by the lower four bits of the UART control register. Below are the possible nibble values for controlling the UART mode:

M3:	M2:	M1:	M0:		Mode:
0	0	0	0		UART off
0	0	0	1		300 Baud
0	0	1	0		1200 Baud
0	0	1	1		2400 Baud
0	1	0	0		4800 Baud
0	1	0	1		9600 Baud
0	1	1	0		19200 Baud
0	1	1	1		38400 Baud
1	0	0	0		57600 Baud
1	0	0	1		115200 Baud
1	0	1	0		230400 Baud
1	0	1	1		9600 Baud
1	1	0	0		9600 Baud
1	1	0	1		9600 Baud
1	1	1	0		9600 Baud
1	1	1	1		9600 Baud

Once the control word is set up in the control register, the UART module is ready to receive and write bytes. To keep track of the state of the UART module, the user can either poll the receive / transmit flags, or the user can set up an interrupt. If the PSoC generates an interrupt on the INTR line back to the 8051, the line will drop low until the user clears the interrupt. The interrupt can be cleared by clearing the transmit and receive flags of the UART module.

Summary of Internal Operation:

The BUS_CLK of the PSoC is set to 48 MHz so that the PSoC can perform its functions quickly enough to complete its task faster than the 8051 can communicate.

The PSoC's Direct Memory Access (DMA) controller is used to emulate memory-mapped registers. Given a command – a read to or write from an address – a DMA channel moves data to and from the appropriate memory location. A byte array labeled '`Reg`' stores the data for the 16 accessible register addresses. A control register is used to write data to the data bus and status registers are used to record bus and address data into the '`Reg`' array and '`Addr`' variable, respectively.

Upon an update to the '`Addr`' variable, the main loop of the PSoC firmware carrys out the appropriate changes to the UART peripheral. Having the main loop control the UART hardware makes the code understandable while the DMA controller provides fixed latency memory accesses to ensure the timing specification is not violated.

4 Example

To demonstrate this 16C450 Replacement, we will attach it to an Amulet module and write some software for the R31JP so that every time a button is pressed on the Amulet module, the character is displayed on the R31JP P1 LED bank as well as on the monitor's display. There are two code examples to demonstrate the polling approach as well as the interrupt approach

After wiring the 8000-series communication portion of this chip to the 8051, attach the PSoC's RX line to the Amulet's TX line. Be sure to connect the ground of the Amulet module to the ground of the R31-JP kit and PSoC. Assemble and load the R31JP with either of the following assembly code examples:

Example 1 – Polling

```
; PSoC 16C450 Replacement Amulet Example - Polling

.equ CTRL_REG, 0fe00h
.equ RX_REG, 0fe02h
.equ RX_FLAG, 40h

.org   000h
ljmp   start

.org   100h
start:
    lcall init
main:
    mov dptr, #CTRL_REG     ; check rx flag
    movx a, @dptr
    anl a, #RX_FLAG
    jz main

    mov dptr, #RX_REG       ; get data
    movx a, @dptr
    mov P1, a
    lcall sndchr

    mov dptr, #CTRL_REG     ; clear the rx flag
    mov a, #05h             ; baud = 9600, no interrupts
    movx @dptr, a
    sjmp  main

init:
; Set up serial communication to the computer
    mov   tmod, #20h        ; set timer 1 for auto reload - mode 2
    mov   tcon, #41h        ; run counter 1 and set edge trig ints
    mov   th1,  #0fdh       ; set 9600 baud with xtal=11.059mhz
    mov   scon, #50h        ; set serial control reg for 8 bit data
                            ; and mode 1

    mov   dptr, #CTRL_REG   ; Set up PSoC UART
    mov   a, #05h           ; baud = 9600, no interrupts
    movx  @dptr, a

    ret

sndchr:
   clr  scon.1              ; clear the tx  buffer full flag.
   mov  sbuf,a              ; put chr in sbuf
txloop:
   jnb  scon.1, txloop      ; wait till chr is sent
   ret
```

Example 2 - Interrupts

```
; PSoC 16C450 Replacement Amulet Example – Interrupt

.org   000h
ljmp   start

.org   003h
ljmp   isr

.org   100h
start:
     lcall init
main:
     sjmp  main


init:
; Set up serial communication to the computer
     mov   tmod, #20h          ; set timer 1 for auto reload - mode 2
     mov   tcon, #41h          ; run counter 1 and set edge trig ints
     mov   th1,  #0fdh         ; set 9600 baud with xtal=11.059mhz
     mov   scon, #50h          ; set serial control reg for 8 bit data
                               ; and mode 1

     mov   IE,   #81h          ; Fully enable the edge triggered interrupt

     mov   dptr, #0xFE00       ; Set up PSoC UART flags for 9600 baud
     mov   a, #0x15            ; communication
     movx  @dptr, a

     ret


isr:
     mov   dptr, #0xFE02       ; Read in the byte from the PSOC
     movx  a, @dptr

     mov   P1, a

     ; Here I send the byte to the PC
     clr  scon.1               ; clear the tx  buffer full flag.
     mov  sbuf,a               ; put chr in sbuf
txloop:
     jnb  scon.1, txloop       ; wait till chr is sent

     mov   dptr, #0xFE00       ; Clear the PSoC UART flags thus clearing
                               ; the external interrupt
     mov   a, #0x15
     movx  @dptr, a

     reti                      ; Return from interrupt
```

Once this code is loaded onto and running on the R31JP, the P1 LED bank and serial interface should respond to presses on the Amulet module.