Assembly Language Programming
Atmel Microprocessors
using the
Linux Operating System

Peter D. Hiscocks
Syscomp Electronic Design Limited
Email: phiscock@ee.ryerson.ca
7 February 2017
Arduino + AVRISP Programmer
Why Atmel?

- Many different parts available: ATTiny to ATXmega.
- Excellent feature set.
- Readily available, reasonable price.
- Large ecosystem: eg, Arduino boards, open-source software, hundreds of hardware shields (interfaces).
- Low and medium complexity units have DIP package.
- Inexpensive development tools: eg, AVRISP $39
- Clean, regular architecture (mostly)
Syscomp Circuitgear Mini uses ATXmega processor.
Typical Arduino Schematic
Why Assembly Language?

Assembly Code Example

USART_Receive:

; Wait for data to be received
in r17, UCSR0A
sbrs r17, RXC
rjmp USART_Receive

; Get and return received data from buffer
in r16, UDR0
ret

C Code Example

unsigned char USART_Receive( void )
{
    /* Wait for data to be received */
    while ( !(UCSR0A & (1<<RXC)) )
;
    /* Get and return received data from buffer */
    return UDR0;
Why Assembly Language (2)?

- For small programs acting as a 'hardware replacement', not much difference between Assembly Language and C.
- Better approach when teaching microprocessor hardware
- Easier understanding code timing issues.
- Simpler programming environment:
  - Assembler vs Compiler-Libraries-Assembler-Linker
- Explicit control over parameter structures in call-return sequence.
Development Process

• Write the program using a text editor: foo.asm
• Assemble the program using an assembler: avra or gavrasm: foo.hex
• Upload foo.hex into the hardware using AVRDUDE program and AVRISP hardware or equivalent.
• Run the program.
• Debug using a serial monitor program.
Hello World for Assembly Language

; Send Character
; This program sends a character out the serial port. The purpose is to
; establish that the microprocessor UART and the computer terminal program
; are configured correctly.
; Terminal program on the host Linux computer: cutecom
; Configuration: 8N1, 9600 baud.
; Reference:
; ATmega168 datasheet, page 237

; Assemble the program.
; Use the AVRISP II programmer to program the Diecimila circuit board
; with the file 'send-char.hex'.
; Connect the USB port on the Diecimila board to the host computer.
; Run cutecom at 8N1, 9600 baud, connected (probably) to ttyUSB0
; Reset the Diecimila board, characters should appear on the terminal.

; Assemble with: gavrasm send-char.asm
; Download with: avrdude -p m168 -c avrisp2 -U flash:w:send-char.hex
; Tested operational 7 March 2017

.DEVICE ATmega168
.CSEG ; strictly speaking not necessary
.ORG 0
    rjmp main ; reset vector points to Main
.ORG 0x100
; stack is not used so SP not initialized.
main:
; Calculate the baud rate constant and set the baud rate
; fosc = 16000000 ; Diecimila crystal oscillator, 16MHz
; baud = 9600
; baudconst = (fosc / (16 x baud) ) -1 ; Calculate the baud constant
.equ baudconst = 103
.equ baudlo = low(baudconst)
equ baudhi = high(baudconst)
Hello World for Assembly Language

; Set the Tx port line PD1 to output
  ldi r16, 0b00000010
  out ddb, r16

; Set the baud rate register
  ldi r16, baudlo
  sts UBRR0L, r16
  ldi r16, baudhi
  sts UBRR0H, r16

; Enable the receiver and transmitter
  ldi r16, 0b00011000
  sts UCSR0B, r16

; Set the frame format: 8 data bits, one stop bit, no parity
  ldi r16, 0b00000110
  sts UCSR0C, r16

; Now send a stream of the same character.
USART_Transmit:
  lds r17, UCSR0A

; Wait for empty transmit buffer
  sbrs r17, UDRE0 ; Skip if bit UDRE is set, transmit is complete
  rjmp USART_Transmit

  ldi r16, "p" ; Send the character
  sts UDR0, r16

  wait: inc r18
  brne wait

; Delay between characters
  rjmp USART_Transmit ; and repeat forever
Hello World for Assembly Language
Serial Monitor for Debugging

- A small program (usually written in assembler) that resides in memory with the program under test.
- Can dump memory locations, test hardware, set breakpoints etc.
- Requires some machine resources: serial port, small amount of memory.
- Can reside in protected memory so it survives reset and reprogramming.
- Communicates with a serial terminal on the host (cutecom).
Alternative Environment

AVR Studio: Windows Only (maybe)

http://www.avrfreaks.net/sites/default/files/HOWTO-AVRStudio%20in%20Ubuntu.pdf
Alternative Environment

C Language Programming
Atmel microprocessors under Linux:

gcc-avr

https://gcc.gnu.org/wiki/avr-gcc