UNIVERSITY OF WALES, ABERYSWYTH
Department of Computer Science

CS25510 - COMPUTER HARDWARE
LABORATORY WORK

THR 68HC11 Simulator - BASIC INPUT/OUTPUT and A/D CONVERSION

1. Using the **editor**, type in the following program:

```
* Automobile Brake Light Program
* Turn on rear brake lights if brake pedal pressed
* Choose PC0 as input from pedal, active high
* Choose PB3,4,5 as outputs to lights, active high

PORTA   EQU    $1000     ; PORTA data register
PORTB   EQU    $1004     ; PORTB data register
PORTC   EQU    $1003     ; PORTC data register
DDRC    EQU    $1007     ; Data Direction register for Port C

BPEDAL  EQU    %00000001 ; Mask for brake pedal bit
BLIGHT  EQU    %00111000 ; Mask for brake light bits

        ORG    $E000          ; Start of ROM (program)

        LDAA   #%00000000     ; Configure all Port C pins
        STAA   DDRC           ; to be inputs

        LDX    #PORTA         ; Point to Port A register
        BCLR   $04,X BLIGHT   ; Initialise lights to be off
MAIN    BRCLR  $03,X BPEDAL   NOBRAKE; If brake pressed, PC0 ==1
        BSET   $04,X BLIGHT   ; Then switch lights on
        BRA    MAIN           ; Test again if brake pressed
NOBRAKE BCLR   $04,X BLIGHT   ; If brake not pressed then lights off
        BRA    MAIN           ; Infinite loop
```

2. **Assemble** and **run** the program, and use the **TH Rijswijk IO Box** for ‘brake light’ output and ‘brake pedal’ input.
3. Examine every line of the program and make sure you understand what operation each **directive** and **instruction** is performing. Go to the **CS25510 home page** and use the Motorola links to help you.
4. Why is #%00000000 stored in **DDRC**? and what is this register?
5. What do the instructions **BCLR**, **BRCLR** and **BSET** do? and what are the roles of **BPEDAL** and **BLIGHT**?
6. What **address mode** is being used to access PORTB and PORTC?
7. Modify the program so that it could be used to deactivate a house alarm system, turn-on the house lights, turn-on the television and turn-off the entry buzzer when an eight bit binary security code is correctly entered via appropriate switches.
8. Further modify the program to use one switch to indicate entry into the house. Upon entry, a buzzer is activated for a set time period until a seven bit binary security code is correctly entered via the remaining switches. Whereupon the program performs those functions detailed in point 7. What should your program do if the security code is not entered correctly within the time limit?

9. Using the **editor**, type in the following program:

```
* Set up 68HC11 A/D.
* Input from Sliders E Port external box.
* Convert analogue input and store in memory address $0000.

        ORG    $0000         ; Start of RAM (data).
RESULT  RMB    1             ; Reserve 1 memory byte for A/D result.

        ORG    $E000         ; Start of ROM (program).
        LDY    #$1000        ; Condition Y register
                             ; to point to OPTION and ADCTL registers.
        BSET   $39,Y $80     ; Set up ADPU in OPTION register
                             ; to power-up A/D.
        BCLR   $39,Y $40     ; Clear bit CSEL in OPTION register
                             ; so that A/D can use E-clock.
AGAIN   LDAB   #$02          ; ACCB = 02.
        STAB   $30,Y         ; Select channel PE2 and clear
                             ; SCAN and MULT bits of the ADCTL
                             ; register. Start conversion process.
CHECKF  LDAA   $1030         ; Read status register ADCTL to see
                             ; if conversion is done.
        BPL    CHECKF        ; If CCF = 0, conversion is in progress,
                             ; go to CHECKF to check again.
        LDAA   $1033         ; If CCF = 1, read the result of the
                             ; A/D conversion from the ADR3 register.
        STAA   RESULT        ; Store the result of the A/D conversion
                             ; of channel PE2 in memory address $0000.
        BRA    AGAIN         ; Infinite loop.
```

10. **Assemble** and **run** the program, and use the **Sliders E Port** to input analogue values for **PE2**. Observe how the memory location **RESULT** changes by using the **Memory List** window.
11. Form the **View** menu, click on **Registers** and then **AD Converter**. Using the AD converter registers window and the CPU registers window, run the program using the **step** button, and observe how the registers change.
12. Refer to the appended notes, p382 - 385, 8.7 PORT E OF THE 68HC11 MCU - A/D CONVERTER, from Tocci's book - Microprocessors and Microcomputers. Using these notes, and from the information you have obtained from point 11 above, examine every line of the program and make sure you understand what operation each **directive** and **instruction** is performing.
13. Modify the program so that it could be used to control a household central heating system. Use one slider to set the desired temperature and use another slider to simulate the actual room temperature. If the actual room temperature falls below the desired temperature value, then turn on an LED on PORTB to indicate activating the central heating. At the same time, turn off another PORTB LED to indicate deactivating the air conditioning. If the actual room temperature rises above the desired temperature value, then turn off the central heating LED on PORTB and turn on the air conditioning LED on PORTB.