

Complete, fold and seal
(see note 2 below)

SURNAME: DEDONCKER

12 Pages

Instructions to Candidates

1. Please give the information requested in the boxes below.
2. Enter your name in the space provided to the right, fold over and seal down (using the label provided) to hide your name.
3. The answers must be legibly written. The Examiners can either ignore illegible script or have it typed and the expense of typing charged to the candidate.
4. Write your answers in ink.
5. Begin each answer on a fresh page. Do all rough work in the answer book and cross it through.
6. No part of this book is to be torn out; it must be handed in complete.
7. If more books than one are used, fasten them together using the tags available from the invigilator(s).
8. Every book must be handed in. In no circumstances may answer books, used or unused, be taken out of the examination venue.

1	Student Number	1203234/2
2	Title of Examination Paper	Configuration and Programming of Embedded Systems
3	Module Code	PSDE SO13
4	Date and Time of Examination	24/5/18 9:30am
5	Examination Venue	Coley Sir Gur Gurug Campus 68250.
6	Seat Number	
7	Number of Answer Books handed in	1

List the numbers of the questions you have attempted	Office Use
Q1	 21.5
a	
b	
c	
Q2	 31.5
a	
b	
c	
d	
Total	57%

$$\frac{52\frac{1}{2}}{102} = 51\%$$

Q 1

a.

Internal CLK = 10 MHz
 $F_{OSC} = 10 \text{ MHz} / 4 = 2.5 \text{ MHz}$
 Timer period = $1 / 2.5 \text{ MHz} = 0.4 \mu\text{s}$
 Timer Prescaler = $0.4 \mu\text{s} \times 256 [\text{prescaler}] = 102.4 \mu\text{s}$
 $10 \mu\text{s} = 10 \mu\text{s} / 102.4 \mu\text{s} = 98$
 errors = $98 - 2 = 96$
 $TMR0 = 256 [\text{prescaler}] - 96 = 160$ ✓

INTCON = 1010 0000

Bit 3 = TOIF Clear

Bit 6 = Peripheral interrupts disabled

Bit 8 = global interrupts enabled. ✓

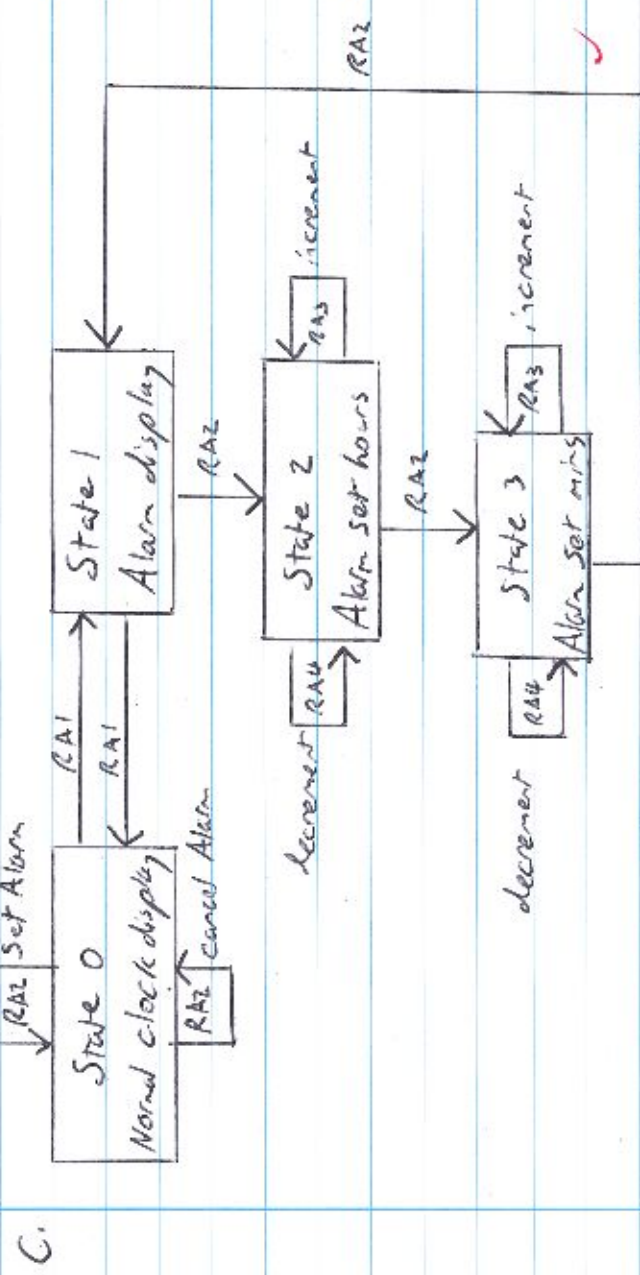
OPTION_REG = 0000 0111

Bit 1-3 = 1; 256 Prescaler maximum

Bit 4 = Enable Prescaler

Bit 6 = TOIC internal clock enable ✓

6. Void interrupt_isr(void) { //begs initial interrupt function
 if (TOIF == 1) { //determines TOIF value
 TOIF = 0; //deactivates TOIF
 TMR0 = 160; //sets new prescaler value
 milli++; //increments milli in steps of +1
 if (milli == 100) { //determines milli value
 milli = 0; //deactivates milli
 secs++; //increments secs in steps of +1
 if (secs == 60) { //determines secs value
 secs = 0; //deactivates secs
 mins++; //increments mins in steps of +1
 if (mins == 60) { //determines mins value
 mins = 0; //deactivates mins
 hrs++; //increments hrs in steps of +1
 if (hrs == 24) { //determines hrs value
 hrs = 0; //deactivates hrs
 TOIF = 1; //sets TOIF to 1, restarts loop
 }
 // ends if hrs == 24
 // ends if mins == 60
 // ends if secs == 60
 // ends if milli == 100
 // ends if TOIF == 1
 }
 // ends main interrupt function ✓
 }
 }
 }
}



Q2.

a. ANO = RAO
 AN1 = RAI
 AN6 = RE1
 AN7 = RE2.

TRISA = 00000011 // defines the bits for RAO and RAI ✓
 TRIS E = 00000110 // defines the bits for RE1 and RE2 ✓
 ADCON0 = 10000001 // defines the bits for A/D Control Register 0 ✓
 ADCON1 = 10000100 // defines the bits for A/D Control Register 1 ✓
 ADCON Channel Bit Selection 5:3 ✓
 ANO = 000
 AN1 = 001
 AN6 = 1110
 AN7 = 1111

8h.

b. Raw value lookup table init:

LUT = ((101-50)+1) = 52 raw elements ✓

The lookup table function allows for new values to be manipulated against the raw values of the ADC. The lookup table stores this data in a linear form. ✓
 stamp value to force linear output.

c. int LUT[52] = {0,0,0,0,1,1,1,1,1,1,2,2,2,2,3,3,3,3,4,4,4,4,5,5,5,5,6,6,6,6,7,7,7,7,8,8,8,8,9,9,9,9,10,10,10,10,11,11,11,12}; ✓

```

d. void main(void) { // Start of main program
    int index, temp, ADC_Value; // declare all variables
    TRISA = 0b00000011; // declare Special Function Registers
    TRISE = 0b000000110; // declare Special Function Registers
    ADCON0 = 0b100000001; // declare Special Function Registers
    ADCON1 = 0b10001000; // declare Special Function Registers
  }

```

```

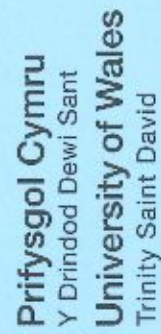
LCD_Startup() { // Start LCD-Startup Function
  while (1) { // Set while loop to 1
    GODONE = 1; // Set GODONE to equal 1.
    while (GODONE == 1); // Set second while loop
    Delay(100); // Delay by 100.
    ADC_Value = (int) ADRESL + (int) ADRESH << 8; // Sets the high and low address results of the ADC.
  }
}

```

```

Display_LCD_Value(1, temp); // Start on LCD
  } // end while loop
} // end LCD-Startup
} // end main program

```



**Complete, fold and seal
(see note 2 below)**

SURNAME: DYER

12 Pages

Instructions to Candidates

1. Please give the information requested in the boxes below.
2. Enter your name in the space provided to the right, fold over and seal down (using the label provided) to hide your name.
3. The answers must be legibly written. The Examiners can either ignore illegible script or have it typed and the expense of typing charged to the candidate.
4. Write your answers in ink.
5. Begin each answer on a fresh page. Do all rough work in the answer book and cross it through.
6. No part of this book is to be torn out; it must be handed in complete.
7. If more books than one are used, fasten them together using the tags available from the invigilator(s).
8. **Every book must be handed in. In no circumstances may answer books, used or unused, be taken out of the examination venue.**

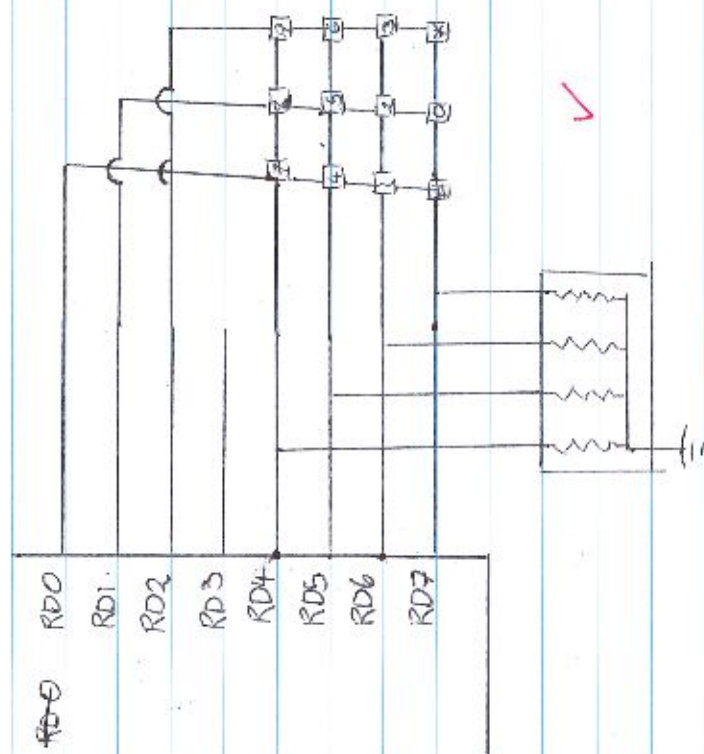
1	Student Number	1503008/1
2	Title of Examination Paper	CONFIG + PROG.EMB, SXT.
3	Module Code	PSDE 5013
4	Date and Time of Examination	24.05.2018 09:30
5	Examination Venue	
6	Seat Number	
7	Number of Answer Books handed in	1

[illegible]

(Gweler yr ochr arall am y fersiwn Cymraeg)

4

(a) A 3×4 matrix encoder works by cross referencing a completed circuit depending on location ~~off~~ of button press in terms of row and column.



Cross-referencing is achieved using if statements, one each for every $i + 1$

if

$$RDO = 1;$$
$$RDI = 0.5$$
$$RD2 = 0;$$
$$RD_4 = 1;$$
$$RDS = 0;$$
$$RD6 = 0;$$
 $RD7 = 0;$
$$\{ \text{Key State} = 73 \}$$

The Cardinals are set than the
"if" statement handles the vovs.

```
void main() { int key-select, set-speed; // variables declared
```

```
void main() {
```

```
key-select = ScanKeypad(); // stores value from keypad  
switch (key-select) {
```

```
case '0': // in the event '0' is entered,
```

```
    Speed(key-select)
```

```
    // '0' is sent forward to
```

```
    set-speed = Speed(key-select) // Speed function
```

```
    key-select = '1';
```

```
case '1': // same for '1'
```

```
    set-speed = Speed(key-select)
```

```
    key-select = '1';
```

```
// code repeats case statements for remaining 2-9
```

```
void Speed (int s)
```

```
    // Function brings in s int
```

```
    int sTu, sHu, sTe, sLn, newSpeed; // variables declared
```

```
    sTu = sHu // values shunted along
```

```
    sHu = sTe
```

```
    sTe = sLn
```

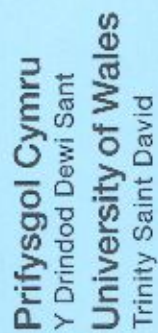
```
    sLn = s
```

```
    // new value added
```

```
    newSpeed = ((sTu * 1000) + (sHu * 100) + (sTe * 1000) + sLn); // values summed
```

```
    if (newSpeed > 9999
```

```
        return newSpeed; // new value for speed returned.
```



**Complete, fold and seal
(see note 2 below)**

SURNAME: Smith

12 Pages

FORENAME(S): Nathan, Sohn

1. Please give the information requested in the boxes below.
2. Enter your name in the space provided to the right, fold over and seal down (using the label provided) to hide your name.
3. The answers must be legibly written. The Examiners can either ignore illegible script or have it typed and the expense of typing charged to the candidate.
4. Write your answers in ink.
5. Begin each answer on a fresh page. Do all rough work in the answer book and cross it through.
6. No part of this book is to be torn out; it must be handed in complete.
7. If more books than one are used, fasten them together using the tags available from the invigilator(s).
8. **Every book must be handed in. In no circumstances may answer books, used or unused, be taken out of the examination venue.**

1	Student Number	11039212
2	Title of Examination Paper	CONFIGURATION + PROGRAMMING EMBEDDED SYSTEMS
3	Module Code	PSDE 5013
4	Date and Time of Examination	24/5/18 9:30
5	Examination Venue	C27
6	Seat Number	
7	Number of Answer Books handed in	1

[illegible]

(Gweler yr ochr arall am y fersiwn Cymraeg)

$$60\frac{1}{2} = 59\frac{1}{2}$$

Question 1

a) Internal CLK = 10MHz FOSC = 10MHz/4 = 2500000Hz; per instruction
 time period = 1/instruction cycle time = 1/2500000 = 0.2us;
 time increment = 0.2us * 256 (prescaler) = 102.4us;
 for 10ms = 10ms / 102.4us = 98;
 98-2 (for errors) = 96;
 preload counter = 256 - 96 = 160;
 TMRO = 160;

CONFIGURATION;

INTCON = 10100000;

Bit 1 and 2 = unused;

Bit 3 = Clear TOIF;

Bit 4 = unused;

Bit 6 = ~~FOSC~~ disable peripheral interrupts;

Bit 8 = global interrupts enable;

OPTION_REG = 00000111

Bit 1-3 = prescaler 1:256 maximum;

Bit 4 = use prescaler enable;

Bit 5 = unused;

Bit 6 = TOSC Select internal clock;

Bit 7 and 8 = unused;

1B)

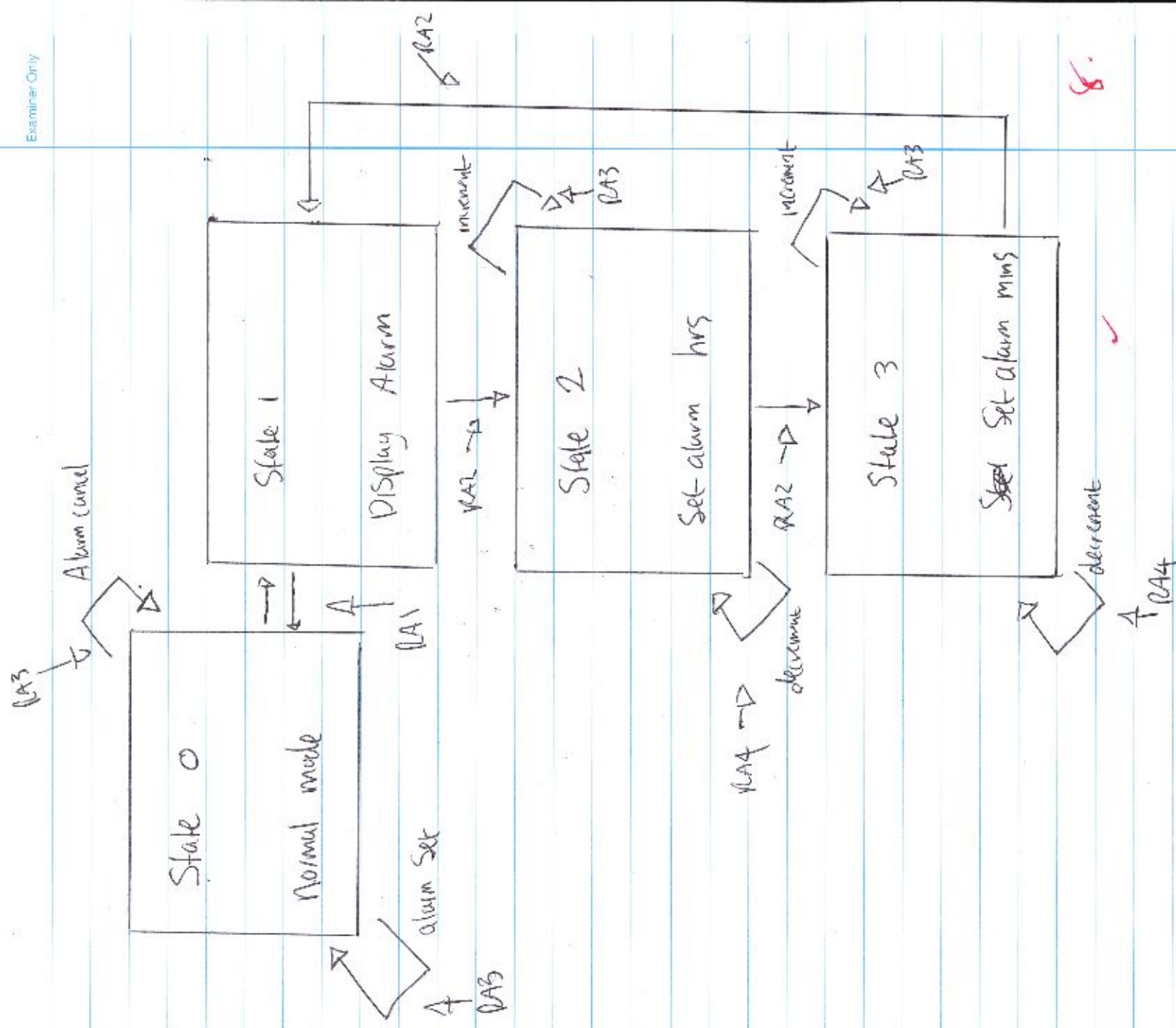
```

void interrupt ISR (void) {
    // no return value
    if (TOIF == 1) {
        TOIF = 0; // Reset interrupt
        TMR0 = 160; // preload for 96 counts
        milli++; // increment milliseconds
        if (milli == 100) { // check for 1 second
            milli = 0; // initialize milliseconds
            secs++; // seconds increment
            if (secs == 60) { // check for 1 minute
                secs = 0; // initialize secs
                mins++; // increment minutes
                if (mins == 60) { // check for 1 hour
                    mins = 0; // initialize mins
                    hrs++; // hours increment
                    if (hrs == 24) { // check for 1 day
                        hrs = 0; // initialize hrs
                        // end if hrs
                    }
                    // end if mins
                }
                // end if secs
            }
            // end if milli
        }
        // end TOIF
    }
    // end ISR
}

```

5/2

1C)



8.

1d) void main(void) {
 TRISA = 0b00011111;
 TRISB = 0b00000000;
 INTCON = 0b10100000;
 OPTION_REG = 0b00001111;
 TMR0 = 160;
 Display_Lcd_Value(4);
 while(1) {
 case 0: DisplayTime(0);
 if (RA) State = 1;
 if (RA3 == 1) {
 if (alarm == 0) {
 alarm = 1;
 }
 }
 else {
 alarm = 0;
 }
 break;
 }
}

→
 while(1) {
 case 0: DisplayTime(0);
 if (RA) State = 1;
 if (RA3 == 1) {
 if (alarm == 0) {
 alarm = 1;
 }
 }
 else {
 alarm = 0;
 }
 break;
 }
}

2a) AN0 = RA0
 AN1 = RA1
 AN6 = RE1
 AN7 = RE2

TRISA = 00000011; // Sets RA0 and RA1 inputs
 TRISE = 00000110; // Sets RE1, RE2 inputs
 ADCON0 = 10000001; // FOSC/32, Enable ADC
 ADCON1 = 10001000; // RA1, RA2 = -VE and +VE

ADCON0 Select value bits 5:3, CHS1, CHS0;

Select Values = AN0 = 000;
 AN1 = 001;
 AN6 = 110;
 AN7 = 111;

2b)

Total number of values stored from the LUT = $((101 - 50) + 1) = 52$ elements

The LUT function is to set or manipulate values from the LUT to represent raw values from the ADC.
 the LUT can manipulate the values to give a linear response by choosing from a number of values on the LUT.

2c)

int (52) = {0, 0, 0, 0, 1, 1, 1, 1, 2, 2, 2, 2, 3, 3, 3, 3, 4, 4, 4, 4, 5, 5, 5, 5, 6, 6, 6, 6, 7, 7, 7, 7, 8, 8, 8, 8, 9, 9, 9, 9, 10, 10, 10, 10, 11, 11, 11, 11, 12, 12, 12, 12}

2d)

```

void main(void) {
    int index, temp, adc_value;
    TRISA = 0b 00000111;
    TRISE = 0b 00000110;
    ADCON0 = 0b 10000001;
    ADCON1 = 0b 10001000;

```

```

    display = lcd_value; // LCD_Start();
    while (1) {
        GO_DONE = 1;
        while (GO_DONE == 1) {
            Delay(100);
            ADC_Value = (int) ADRESL + (int) ADRESH << 8;

```

GO_DONE

```

        index = ADC_Value - 50;

```

```

        temp = LUT[index];

```

```

        display_lcd_value(1, temp);

```

```

    } //end
} //end

```

12.5

12

12.5

Examination Answer Books

12 Pages

Instructions to Candidates

1. Please give the information requested in the boxes below.
2. Enter your name in the space provided to the right, fold over and seal down (using the label provided) to hide your name.
3. The answers must be legibly written. The Examiners can either ignore illegible script or have it typed and the expense of typing charged to the candidate.
4. Write your answers in ink.
5. Begin each answer on a fresh page. Do all rough work in the answer book and cross it through.
6. No part of this book is to be torn out; it must be handed in complete.
7. If more books than one are used, fasten them together using the tags available from the invigilator(s).
8. Every book must be handed in. In no circumstances may answer books, used or unused, be taken out of the examination venue.

Complete, fold and seal
(see note 2 below)

SURNAME: Jones

FORENAME(S): Joseph Owen

1	Student Number	1502999/1
2	Title of Examination Paper	Config + Plug Embedded Systems
3	Module Code	PSDE5013
4	Date and Time of Examination	24.5.18 9:30am
5	Examination Venue	C27
6	Seat Number	
7	Number of Answer Books handed in	1

List the numbers of the questions you have attempted	Office Use
Q1	22
Q2	19
Q4	0
Total	40%

a) internal CLK = 10MHz $f_{osc} = 10\text{MHz} / 4 = 2500000\text{Hz}$ per instruction

Time period = $1 / f_{osc} = 1 / 2500000 = 0.4\mu\text{s}$

Timer increment = $250.4\mu\text{s} * 256 (\text{Prescaler}) = 102.4\mu\text{s}$

For 10ms = $10\text{m} / 102.4\mu\text{s} = 98$ ✓

98 - 2 for zeros = 96 ✓

Preload for ~~TMRB~~ ^{TMRD} = $256 - 96 = 160$ ✓

TMRD = 160 // code used for 10ms ✓

Configuration

INTCON = 10100000 ✓

Bit 1 and 2 - unused

Bit 3 - clear T0IF

Bit 4 - unused

Bit 6 - disable peripheral interrupts

Bit 8 - enable global interrupts ✓

OPTIONS - REG = 0000111 ✓

Bits 1-3 - Prescaler - 1:256 maximum

Bit 4 - enable prescaler

Bit 5 - unused

Bit 6 - T0SC for chosen internal clock

Bits 7 and 8 - unused ✓

b)

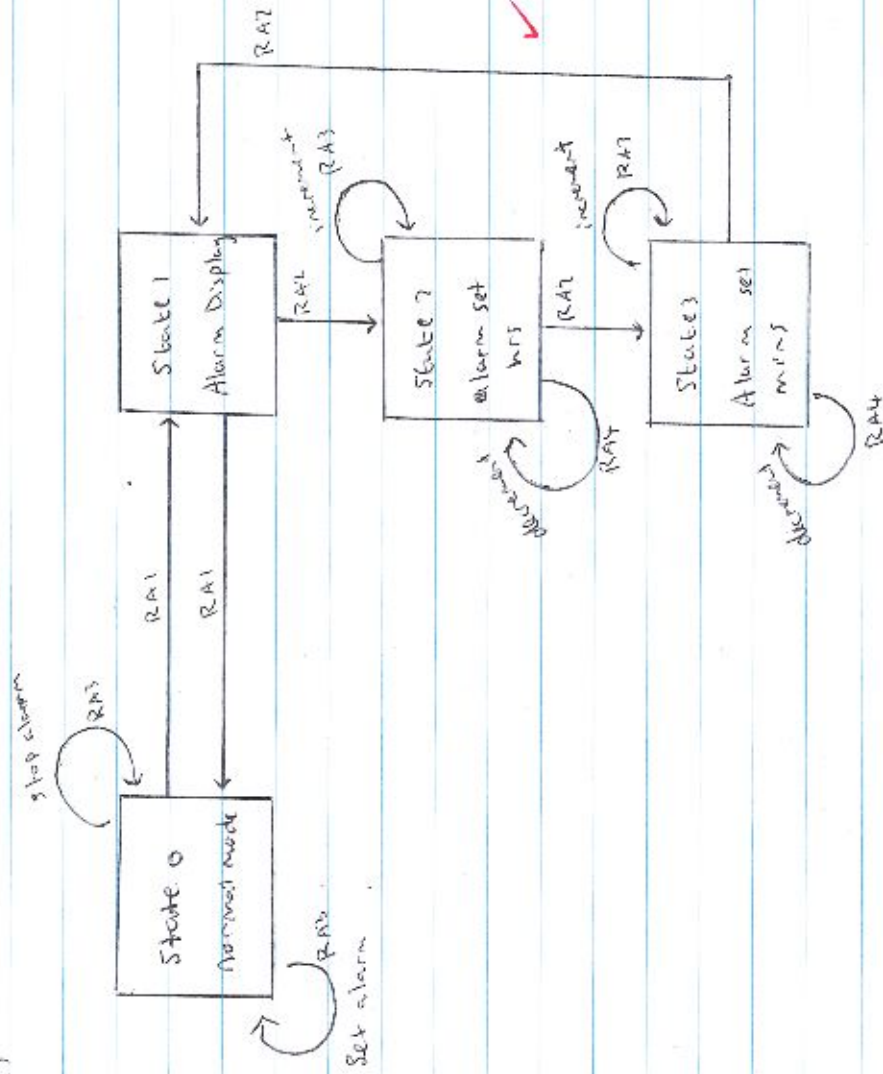
```

void interrupt ISR (void) { // no value in function ✓
    if (TOIF == 1) { // if TMR0 has an interrupt ✓
        TOIF = 0; // reset interrupt ✓
        TMR0 = 160; // preload for TOIF - 10ms ✓
        milli++; // increment milliseconds x 10 ✓
        if (milli == 100) { // we if 1 second ✓
            milli = 0; // initialise milli ✓
            secs++; // increment secs ✓
            if (secs == 60) { // if 1 minute ✓
                secs = 0; // initialise secs ✓
                mins++; // increment minutes ✓
                if (mins == 60) { // if 1 hour ✓
                    mins = 0; // initialise mins ✓
                    hrs++; // increment hour ✓
                    if (hrs == 24) { // when 1 day ✓
                        hrs = 0; // initialise hours ✓
                    } // end if hrs
                } // end if mins
            } // end if secs
        } // end if milli
    } // end if TOIF
} // end interrupt ISR ✓

```

5/6

c)



Q.

AND = RAO

$$A_{N-1} = R_{A-1}$$
$$\text{ANG} = \text{REI}$$
$$AN7 = RE2$$

TRISA = 00000011 // 2A set RA0, RA1 as inputs ✓

TRISE = 0000110 ✓ // set REU, REZ as inputs ✓

ADcno - 1000001 ✓ // Probable ADC, Fosc 12, RA0/AN0 ✓

$AD_{CON1} = 1000$ ✓ right justified ✓

$$-4 V_L R_A^2 + V_L R_E^2$$
$$A_{\text{NO}} = 0.00$$

AN1 = 001

1101

AW7 = 111

AlOCONO channel select brist are

§ 13 CHS 2 CHS 1 CHS 0

b) There are 52 elements because of $((101 - 50) + 1) = 52$. The look up table function is to help linearize a device's throughput by normalizing the input temp value.

c) in $\mathbb{N}$ $L_{WT} = \{0, 0, 0, 0, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 3, 3, 3, 3, 3, 4, 4, 4, 4, 4, 5, 5, 5, 5, 5, 6, 6, 6, 6, 6, 7, 7, 7, 7, 7, 8, 8, 8, 8, 8, 8, 9, 9, 9, 9, 10, 10, 10, 10, 10, 11, 11, 11, 11, 12, 12, 12, 12, 12\}$ ✓

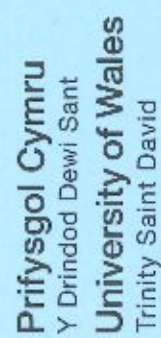
```

d) void LCD_startwd();

```

Variable Display - LCD - Value (in line, interval)

No id delivery (int f) ~~X~~



**Complete, fold and seal
(see note 2 below)**

SURNAME: BÉLÉ

FORENAME(S): JOSHUA

12 Pages

Instructions to Candidates

1. Please give the information requested in the boxes below.
2. Enter your name in the space provided to the right, fold over and seal down (using the label provided) to hide your name.
3. The answers must be legibly written. The Examiners can either ignore illegible script or have it typed and the expense of typing charged to the candidate.
4. Write your answers in ink.
5. Begin each answer on a fresh page. Do all rough work in the answer book and cross it through.
6. No part of this book is to be torn out; it must be handed in complete.
7. If more books than one are used, fasten them together using the tags available from the invigilator(s).
8. **Every book must be handed in. In no circumstances may answer books, used or unused, be taken out of the examination venue.**

1	Student Number	120326212
2	Title of Examination Paper	Config. Prog. , Em6 systems
3	Module Code	PSDE 5013
4	Date and Time of Examination	24/5/18 9:30 - 11:30
5	Examination Venue	
6	Seat Number	
7	Number of Answer Books handed in	

[illegible]

(Gweler yr ochr arall am y fersiwn Cymraeg)

1a) unemul CLK = 10 MHz FOSC = 10 MHz / 4 = 2500000 Hz per ~~un~~ instruction
 time period = 1 / instruction cycle time = 1 / 2500000 = 0.4 μ s
 timer increment = 0.4 μ s / 256 [prescaler] = 102.4 μ s
 for 10ms = 10ms / 102.4 μ s = 98
 98 - 2 for errors = 96
 preloaded counter TMR0 = 256 - 96 = 160
 TMR0 = 160 // in code for 10ms delay ✓

configuration:

INTCON = ~~00000000~~ 10100000

bits 1 and 2 - unused

bit 3 - clear TOIF

bit 4 - unused

bit 6 - disable peripheral interrupt

bit 8 - enable global interrupt ✓

OPTION_REG = ~~10000000~~ 00000111 ✓

bits 1-3 - prescaler set to 1:256 maximum ✓

bit 4 - ~~enable~~ prescaler used enabled ✓

bit 5 - unused

bit 6 - TOSC internally clock selected ✓

bit 7 and 8 - unused

Question No.	Examiner Only	Question No.	Examiner Only
1d)	<pre> void main (void) { // setup main program TRISA = 0b00001111; // set alarm as outputs and switches as inputs TRISB = 0b00000000; // set LCD as outputs JNTCON = 10100000; // set up interrupts OPTION_REG = 0b00001111; // set up prescaler 1:256 TMRO = 160; // reload TMRO to 160 are count of 6 LCD_Startup(); // power up LCD and initialise } while (1) { // setup super loop case 0: DisplayTime (0); // normal mode if (RA1) state = 1; // go to state 1 if (RA3 = 1) { if (alarm = 0) { // alarm is off alarm = 1; // set alarm } } else { alarm = 0; // cancel alarm } break; case 1: Display AlarmTime (0); // Display Alarm on LCD time 0 if (RA1) state = 0; // normal mode if (RA2) state = 2; // go to state 2 break; case 2: Display Set AlarmTime (0); // Display Alarm set time on LCD time 0 if (RA1) state = 1; // go to state 1 if (RA2) state = 3; // go to state 3 if (RA3) { alarm_hrs++; // increment hrs } if (RA4) { alarm_hrs--; // decrement hrs } break; } } </pre>	1d) <pre> case 3: Display Set AlarmTime (0); // Display alarm set time LCD time 0 if (RA1) state = 1; // go to state 1 if (RA2) state = 0; // go to normal mode if (RA3) { alarm_hrs++; // increment hrs } if (RA4) { alarm_hrs--; // decrement hrs } break; } // end while // end main </pre> 2a) <pre> AN0 = RA0 AN1 = RA1 AN6 = RE1 AN7 = RE2 </pre> <pre> TRISA = 0b00000111; // set RA0 and RA1 as inputs TRISE = 0b00001110; // set RE1 and RE2 as inputs ADCON0 = 10000001; // FOSC/32, set ADC on and channel 1 selected ADCON1 = 10001000; // bit 7 right justified, 6 channels // RA2 and RA3 VREF -ve and +ve ADCON0 selecting channels bits are 5:3 CHS2, CHS1, CHS0 select channels AN0 = 000 AN1 = 001 AN6 = 110 AN7 = 111 </pre> b) 52 elements used because top Range ADC value (101) - bottom range value (50) + 1 = 52 The LUT function is to create more linear results from the ADC value by assigning them to the unit values. This works because more than 1 value of ADC can be assigned to 1 unit value to make the scale linear.	11/2 ✓
Answer yn unig	Phil Y Question	Answer yn unig	Phil Y Question

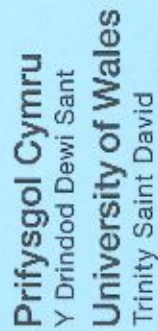
c) ~~array = [0,0,0,0,1,1,1,1,2,2,2,2,3,3,3,3,4,4,4,4,4,4,5,5,5,5,6,6,6,6,7,7,7,7,8,8,8,8,9,9,9,9,10,10,10,10,11,11,11,12];~~
 array = [0,0,0,0,1,1,1,1,2,2,2,2,3,3,3,3,4,4,4,4,4,4,5,5,5,5,6,6,6,6,7,7,7,7,8,8,8,8,9,9,9,9,10,10,10,10,11,11,11,12];

d) void main(void) { // start of main program
 int index, temp, ADC_Value; // declare variables
 TRISA = 0b00000001;
 TRISE = 0b00000010;
 ADCON0 = 0b1000001; // A/D converter 300 and FOSC/32
 ADCON1 = 0b10001000; // most sig bit of ADRESH else read as 0

while (1) { // start single loop
 GDDONE = 1;
 while (GDDONE == 0);
 Delay(100); // set delay of 100
 temp = (int) ADRESL + (int) ADRESH << 8; ✓

int index = ADC_Value - 50; // measuring index value = to ADC value
 temp = LUT[index]; ✓ // storing LUT temp in index array
 LCD_Display(LCD_Value, (1, temp)); // displaying temperature on LCD

} // end while
 } // end main ✓



Complete, fold and seal
(see note 2 below)

SURNAME: HARRIES

FORENAME(S): DAVID SAMUEL

12 Pages

Instructions to Candidates

1. Please give the information requested in the boxes below.
2. Enter your name in the space provided to the right, fold over and seal down (using the label provided) to hide your name.
3. The answers must be legibly written. The Examiners can either ignore illegible script or have it typed and the expense of typing charged to the candidate.
4. Write your answers in ink.
5. Begin each answer on a fresh page. Do all rough work in the answer book and cross it through.
6. No part of this book is to be torn out; it must be handed in complete.
7. If more books than one are used, fasten them together using the tags available from the invigilator(s).
8. **Every book must be handed in. In no circumstances may answer books, used or unused, be taken out of the examination venue.**

1	Student Number	1502997-1
2	Title of Examination Paper	CONFIG AND PROGRAMMING-EMBEDDED SYSTEMS
3	Module Code	PSPE 5013
4	Date and Time of Examination	24/5/18 9:30
5	Examination Venue	
6	Seat Number	
7	Number of Answer Books handed in	

[illegible]

$$\frac{61\frac{1}{2}}{102} = 60\%$$

Q2 a)	AND = RAO;
	ANI = RAI;
	ANB = REI;
	AN7 = RE2;

	✓	11. Ber	reference
TRISA = 0b00000011;	✓		
TRISE = 0b00000110;	✓	4	
ADCON0 = 10000001;	✓	4	2
ADCON1 = 10001000;	✓	4	

AND = 000
AN1 = 001
AN6 = 110
AN7 = 111

b) Number of elements in ARRAY = $((101 - 50) + 1) = 52$ elements ✓

The LUT (look up table) is used to artificially create a linear output from the ADC. The LUT assigns the raw value of 50 to 0° and a value of 101 for 120°. The temperature values inbetween are assigned to ADC values creating a linear relationship. ✓

~~DATE~~ = [0606] + [2222] = 0622


```

IF (RA4 == 1)
{
    state = 1;
    RA4 = 0;
    // break;
}
// end while
// end program

```

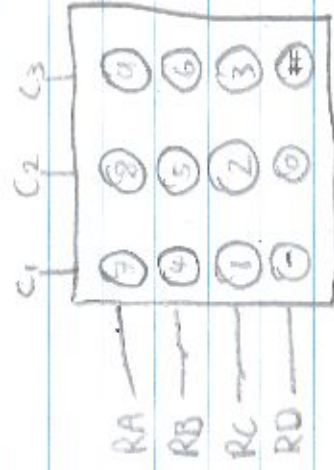
The resets on RA4, RA3, RA2, RA1 are not needed but must affect the operation.

The entry read error found in no "break;" after "Case 3"

Q4) a)

The ports D pins are used on the microcontroller to connect the keypad.

3 pins are used for the columns (RD0, RD1, RD2)



4 pins are used for the rows (RD4, RD5, RD6, RD7). This means that 7 out of 8 port D pins are used to interface with the keypad. The microcontroller determines what buttons are pressed through the use of a multiplex function as shown below.

```

int ScanKeypad (Key, keypad_enable)
{
    // Key = -1;
    // Set Key to -1

```

```

    // column 1 (C1)
    // Row 1 (RA)

```

```

    IF (RD4 == 1) Key = 7;

```

```

    IF (RD5 == 1) Key = 4;

```

```

    IF (RD6 == 1) Key = 1;

```

```

    IF (RD7 == 1) Key = -1;

```

RD0 = 0;

```

    RD1 = 1;

```

```

    IF (RD4 == 1) Key = 8;

```

```

    IF (RD5 == 1) Key = 5;

```

```

    IF (RD6 == 1) Key = 2;

```

```

    IF (RD7 == 1) Key = 0;

```

RA1 = 0;

```

    RD2 = 1;

```

```

    IF (RD4 == 1) Key = 9;

```

```

    IF (RD5 == 1) Key = 6;

```

```

    IF (RD6 == 1) Key = 3;

```

```

    IF (RD7 == 1) Key = #;

```

```

    // return Key;

```

} return determined Key value