Universiti Tun Hussein Onn Malaysia

UNIVERSITI TUN HUSSEIN ONN MALAYSIA

FINAL EXAMINATION
SEMESTER I
SESSION 2016/2017

COURSE NAME	:	REAL TIME EMBEDDED SYSTEM
COURSE CODE	:	BEH 30802
PROGRAMME CODE	:	BEJ
EXAMINATION DATE	:	DECEMBER 2016 / JANUARY 2017
DURATION	:	2 HOURS 30 MINUTES
INSTRUCTION	:	ANSWERS ALL QUESTIONS

THIS QUESTION PAPER CONSISTS OF **FIVE (5)** PAGES

Q1 (a) Give a definition of real-time system.

(2 marks)

(b) Explain **two (2)** challenges of embedded system for real-time system application.

(4 marks)

(c) Distinguish the key differences between hard and soft real time systems by providing an example for each one.

(4 marks)

Q2 (a) An LED and a button switch are connected to Arduino Uno standard platform.

(i) Sketch a schematic for LED connection to pin D4 of microcontroller that based on sinking mode.

(3 marks)

(ii) Sketch a schematic for button switch connection to pin D5 of microcontroller that based on pull-down resistor concept.

(3 marks)

(iii) Write a complete Arduino code for controlling the LED based on the button switch state.

(8 marks)

(b) Real time clock (RTC) is an external slave device that can be connected to microcontroller using serial communication either serial peripheral interface (SPI) or inter-integrated circuit (I2C) protocol. Compare between SPI and I2C protocol in term of connection and baud rate (communication speed).

(6 marks)

Q3 (a) List **four (4)** features of Real-Time Operating System.

(2 marks)

(b) Discuss **two (2)** type of schedulers in Real-Time Operating System.

(4 marks)

(c) The hardware system same as **Q2(a)** is used and its software is modified to handle three threads by using ChibiOS/RT. The code of the operation is given as follows:

```
#include <ChibiOS_AVR.h>
unsigned char BTN;

//Thread 1
// to be completed as in Q3(c)(i) instruction.

//Thread 2
static WORKING_AREA(waT2,64);
static msg_t Th2(void *arg)

{
 while(1)
 {
  digitalWrite(4,BTN);
  chThdSleepMilliseconds(50);
 }
 return 0;
}

//Thread 3
static WORKING_AREA(waT3,64);
static msg_t Th3(void *arg)
{
 while(1)
 {
  Serial.print(BTN);
  chThdSleepMilliseconds(100);
 }
 return 0;
}

void setup()
{
 Serial.begin(9600);
 pinMode(3,INPUT);
 pinMode(4,OUTPUT);
 chBegin(chSetup);
}

void chSetup()
{
 chThdCreateStatic(waT1,sizeof(waT1),NORMALPRIO+2,Th1,NULL);
 chThdCreateStatic(waT2,sizeof(waT2),NORMALPRIO+1,Th2,NULL);
 chThdCreateStatic(waT3,sizeof(waT3),NORMALPRIO,Th3,NULL);
}

void loop() { }
```

(i) Complete a task function for Thread1 operation by named it as Th1. Its working area size is 128 bytes. This task will be executed every 25ms (period) for reading a status of button switch and save in 'BTN' variable.

(4 marks)

(ii) Give detail explanation on the output prediction of the system operation.

(8 marks)

(iii) Provide effective solutions for reducing the memory usage and power consumption.

(3 marks)

Q4 (a) Explain the following terminology in resource sharing application.

(i) Semaphore

(2 marks)

(ii) Mutex

(2 marks)

(iii) Priority Inversion

(2 marks)

(b) Analyse the operation of code in **Q3(c)** in term of deadlock, starvation and priority inversion events.

(6 marks)

(c) The coding in **Q3(c)** is going to add a counting semaphore for synchronising 'BTN' variable from Thread1 to Thread2 and Thread3.

(i) Modify the existing code for declaring the semaphore token, wait and signal operation.

(6 marks)

(ii) Propose two solutions to avoid deadlock condition when using semaphore.

(4 marks)

Q5 (a) Define the deadline (DL) and maximum elapsed time (Max_E) of a task in temporal scope.

(2 marks)

(b) Assume a system has three independent tasks A, B, and C with periods of 40, 15, and 30 ms and CPU time of 10, 5, and 5ms respectively.

(i) If the priority level of Task A > Task B > Task C, draw a task activition diagram for the first 100ms of system operation.

(6 marks)

(ii) Create a table to state the start delay, elapse time and completion time for each task.

(6 marks)

(c) Another way to analyze the schedulability of the tasks in the **Q5(b)** is by using full test of rate monotonic schedulability (RMS).

(i) Re-arrange the new priority level for each task that based on RMS concept.

(2 marks)

(ii) Calculate the worst-case completion time for each task.

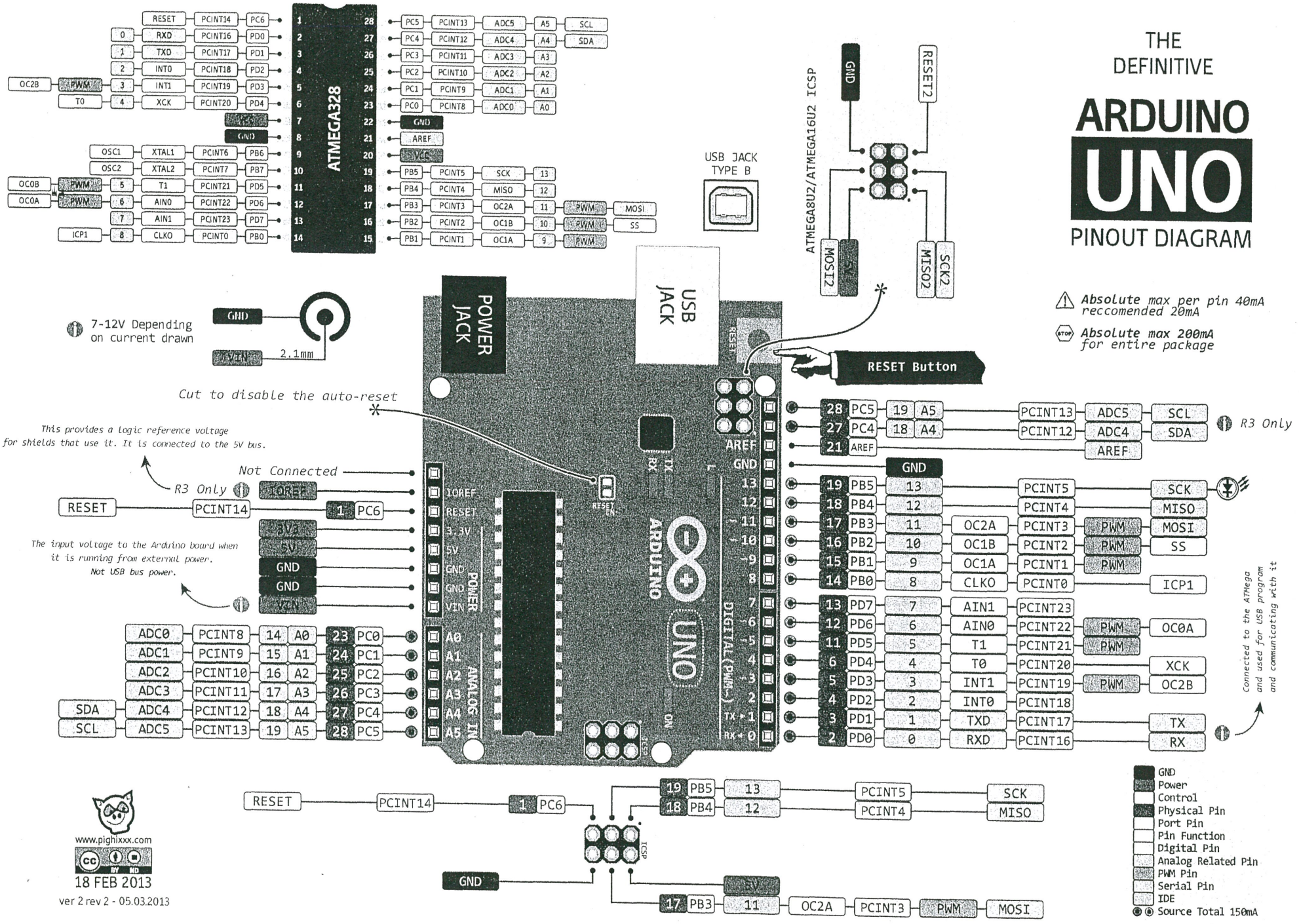
(8 marks)

(iii) Comment on the ability of each task to meet its deadline.

(3 marks)

-END OF QUESTIONS –

THE DEFINITIVE ARDUINO UNO PINOUT DIAGRAM



www.pighixxx.com

18 FEB 2013

ver 2 rev 2 - 05.03.2013

TERBUKA

Arduino Programming Cheat Sheet

Primary source: Arduino Language Reference
http://arduino.cc/en/Reference/

TERBUKA

Structure & Flow

Basic Program Structure

```
void setup() {
  // runs once when sketch starts
}
void loop() {
  // runs repeatedly
}
```

Control Structures

```
if (x < 5) { ... } else { ... }
while (x < 5) { ... }
do { ... } while ( x < 5);
for (int i = 0; i < 10; i++) { ... }
break;  // exit a loop immediately
continue;  // go to next iteration
switch (myVar) {
   case 1:
     ...
     break;
   case 2:
     ...
     break;
   default:
     ...
}
return x; // just return; for voids
```

Operators

General Operators

```
=   (assignment operator)
+   (add)          -   (subtract)
*   (multiply)     /   (divide)
%   (modulo)
==  (equal to)     !=  (not equal to)
<   (less than)    >   (greater than)
<=  (less than or equal to)
>=  (greater than or equal to)
&&  (and)      ||  (or)      !  (not)
```

Compound Operators

```
++  (increment)
--  (decrement)
+=  (compound addition)
-=  (compound substraction)
*=  (compound multiplication)
/=  (compound division)
&=  (compound bitwise and)
|=  (compound bitwise or)
```

Bitwise Operators

```
&  (bitwise and)     |  (bitwise or)
^  (bitwise xor)     ~  (bitwise not)
<< (shift left)     >> (shift right)
```

Built-in Functions

Pin Input/Output

```
Digital I/O (pins: 0-13 A0-A5)
  pinMode(pin,[INPUT, OUTPUT])
  int digitalread(pin)
  digitalWrite(pin, value)
    // Write HIGH to an input to
    // enable pull-up resistors
Analog In (pins: 0-5)
  int analogRead(pin)
  analogReference(
    [DEFAULT, INTERNAL, EXTERNAL])
PWM Out (pins: 3 5 6 9 10 11)
  analogWrite(pin, value)
```

Advanced I/O

```
tone(pin, freqhz)
tone(pin, freqhz, duration_ms)
noTone(pin)
shiftOut(dataPin, clockPin,
  [MSBFIRST,LSBFIRST], value)
unsigned long pulseIn(pin,
  [HIGH,LOW])
```

Time

```
unsigned long millis()
  // overflows at 50 days
unsigned long micros()
  // overflows at 70 minutes
delay(msec)
delayMicroseconds(usec)
```

Math

```
min(x, y)    max(x, y)    abs(x)
sin(rad)     cos(rad)     tan(rad)
sqrt(x)      pow(base, exponent)
constrain(x, minval, maxval)
map(val, fromL, fromH, toL, toH)
```

Random Numbers

```
randomSeed(seed) // long or int
long random(max)
long random(min, max)
```

Bits and Bytes

```
lowByte(x)     highByte(x)
bitRead(x, bitn)
bitWrite(x, bitn, bit)
bitSet(x, bitn)
bitClear(x, bitn)
bit(bitn)  // bitn: 0=LSB 7=MSB
```

Type Conversions

```
char()      byte()
int()       word()
long()      float()
```

External Interrupts

```
attachInterrupt(interrupt, func,
 [LOW, CHANGE, RISING, FALLING])
detachInterrupt(interrupt)
interrupts()
noInterrupts()
```

Variables, Arrays, and Data

Data types

```
void
boolean   (0, 1, true, false)
char      (e.g. 'a'  -128 to 127)
int       (-32768 to 32767)
long      (-2147483648 to 2147483647)
unsigned char  (0 to 255)
byte           (0 to 255)
unsigned int   (0 to 65535)
word           (0 to 65535)
unsigned long  (0 to 4294967295)
float     (-3.4028e+38 to 3.4028e+38)
double    (currently same as float)
```

Qualifiers

```
static     (persists between calls)
volatile   (in RAM (nice for ISR))
const      (make read only)
PROGMEM    (in flash)
```

Arrays

```
int myInts[6]; // array of 6 ints
int myPins[]={2, 4, 8, 3, 6};
int mySensVals[6]={2, 4, -8, 3, 2};
myInts[0]=42;  // assigning first
               // index of myInts
myInts[6]=12;  // ERROR! Indexes
               // are 0 though 5
```

Constants

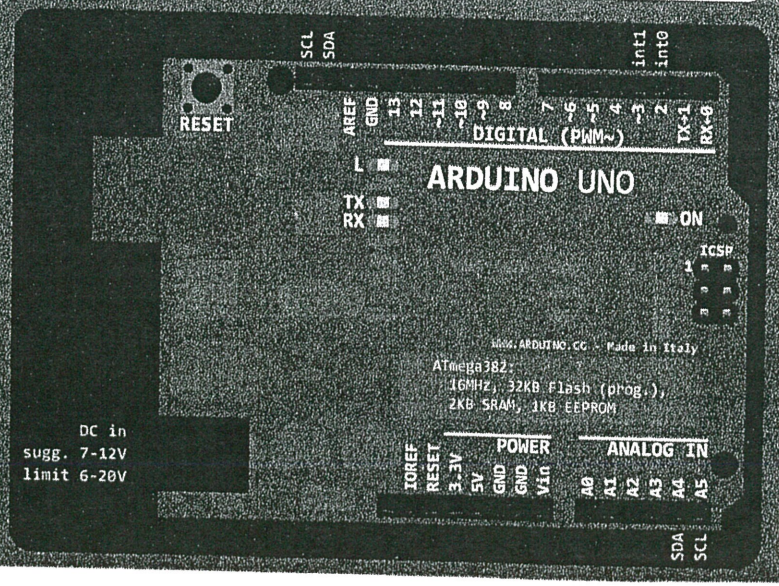
```
HIGH  | LOW
INPUT | OUTPUT
true  | false
143          (Decimal)
0173         (Octal - base 8)
0b11011111   (Binary)
0x7B         (Hexadecimal - base 16)
7U           (force unsigned)
10L          (force long)
15UL         (force long unsigned)
10.0         (force floating point)
2.4e5        (2.4*10^5 = 240000)
```

Pointer Access

```
&  (reference: get a pointer)
*  (dereference: follow a pointer)
```

Strings

```
char S1[8] =
 {'A','r','d','u','i','n','o'};
 // unterminated string; may crash
char S2[8] =
 {'A','r','d','u','i','n','o','\0'};
 // includes \0 null termination
char S3[]="Arduino";
char S4[8]="Arduino";
```



Libraries

Serial (communicate with PC or via RX/TX)

```
begin(long Speed) // up to 115200
end()
int available() // #bytes available
byte read() // -1 if none available
byte peek()
flush()
print(myData)
println(myData)
write(myBytes)
SerialEvent() // called if data rdy
```

SoftwareSerial (serial comm. on any pins)

```
  (#include <softwareSerial.h>)
SoftwareSerial(rxPin, txPin)
begin(long Speed) // up to 115200
listen()        // Only 1 can listen
isListening() // at a time.
read, peek, print, println, write
  // all like in Serial library
```

EEPROM (#include <EEPROM.h>)

```
byte read(intAddr)
write(intAddr, myByte)
```

Servo (#include <Servo.h>)

```
attach(pin, [min_uS, max_uS])
write(angle)  // 0 to 180
writeMicroseconds(uS)
   // 1000-2000; 1500 is midpoint
int read()    // 0 to 180
bool attached()
detach()
```

Wire (I²C comm.) (#include <Wire.h>)

```
begin()       // join a master
begin(addr) // join a slave @ addr
requestFrom(address, count)
beginTransmission(addr) // Step 1
send(myByte)            // Step 2
send(char * mystring)
send(byte * data, size)
endTransmission()       // Step 3
int available() // #bytes available
byte receive()  // get next byte
onReceive(handler)
onRequest(handler)
```