The Open University of Sri Lanka
Faculty of Engineering Technology
Department of Electrical and Computer Engineering

Study Programme	: Bachelor of Software Engineering Honours
Name of the Examination	: Final Examination
Course Code and Title	**: EEX5376 Embedded Systems and IoT**
Academic Year	: 2024/25
Date	: 09th August 2025
Time	: 13:30-16:30hrs
Duration	: **3 hours**

General Instructions

1. Read all instructions carefully before answering the questions.

2. This question paper contains one (01) question in SECTION A and two (2) questions in SECTION B on **five (5)** pages.

3. Answer all questions in SECTION A. [60 Marks], and SECTION B. [40 Marks].

4. Answer for each question should commence on a new page.

5. Refer to the Annexure of the Arduino programming syntax given on page four (4) to write an Arduino programming code.

6. This is a Closed Book Test (CBT).

7. Answers should be in clear handwriting.

8. Do not use Red colour pen, and clearly state your assumptions if any.

Section A: Answer ALL questions [60 Marks]

The following description is about the **Smart Traffic and Road Safety Management System (STRSMS)**. Your task is to analyse the following specifications and design the **STRSMS** by providing IoT solutions for the STRSMS.

Every city wants fewer road traffic accidents, especially major ones. So, the solution employs state-of-the-art technology to quickly detect, analyse, and handle vehicle violations, effectively maintaining and promoting greater traffic order.

It's also built to improve and accelerate the handling of any accidents that do occur. The solution detects and warns of accidents within 10s of them happening, immediately communicating with the emergency handling system. It then quickly releases guidance information based on traffic volumes. Quite simply, such an End to End (E2E) approach to incident handling makes travel far safer for all.

The data-based Intelligent Transportation System (ITS) Solution applies image vision, the Internet of Things (IoT), Artificial Intelligence (AI), big data, and other technologies to traffic management. Adopting unified access standards, it centrally manages aggregated data to better serve upper-layer service applications such as violation handling, vehicle analysis, and vehicle management and control, as well as traffic planning decision-making.

[Reference:https://e.huawei.com/en/industries/urban-traffic/urban-traffic-management/intelligent-transportation-system]

[Q1]

i) Draw an edge-fog-cloud IoT architecture diagram for the proposed STRSMS solution and provide a brief explanation. Clearly indicate what the edge devices, gateways, fog devices, connectivity, cloud services, and applications are. [16 Marks]

ii) Draw the following diagrams for the 3-way traffic light control system with two modes, automatic operation (dynamically adapting the timing by sensing the traffic density) and manual traffic lights control operation of the STRSMS solution. Briefly explain each diagram. [32 Marks]

(a) Process specification diagram.
(b) Domain model specification diagram.
(c) Information model specification diagram
(d) Service specification diagram

iii) Refer to the Arduino programming instructions and write a program for the [Q1] ii) process. State the comments where necessary. [12 Marks]

SECTION B: Answer all the questions. [40 Marks]

[Q2]

(i) In the context of the emergency handling system within the STRSMS solution, write a program using pseudo-code for two algorithms,

(a) Dispatching an available ambulance to an accident site.

(b) Select the most suitable hospital and transport the patient efficiently.

(*Hint: you should consider minimising response time, selecting the nearest available ambulance, optimising travel time to the accident site, and choosing the most appropriate hospital with the shortest travel duration*).

[12 Marks]

(ii) Briefly explain four (04) security concerns related to the proposed STRSMS solution and the steps that could be taken to mitigate them. [8 Marks]

[Q3]

(i) List three (03) lightweight data transfer techniques with short descriptions for your proposed STRSMS solution. [6 Marks]

(ii) Compare and contrast the MQTT and XMPP data transfer protocols and justify the data transfer protocol selection for the proposed STRSMS solution.

[6 Marks]

(iii) Draw the block diagram and briefly describe the distributed view of the data processing architecture of the proposed STRSMS IoT application. (*Hint: Should discuss the types of data used in each level or layer, type of data analysis, and level or layer where it is applied*)

[8 Marks]

-------- END--------

Annexure

Syntax of selected instructions of the Arduino programming

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) {……}
for ( int i = 0; i< 10; i++) {……}
break;       // Exit a loop immediately
continue;    // Go to next iteration
switch (var) {
case 1:
  ……
break;
   case 2:
 ……..
break;
   default:
 ……..
}
return x;    // x must match return type
return;      // For void return type
```

Function Definitions

<ret. type><name> (params) {……}
e.g. int double (int x) {return x*2;}

Operators

General operators

= assignment
+ 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 subtraction
*= compound multiplication
/= compound division
&= compound bitwise and
|= compound bitwise or

Bitwise Operators

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

Pointer Access

& reference: get a pointer
* dereference: follow a pointer

Variables, Arrays, and Data

Data types

bool	true	|	false
char	-128	-	127, 'a' '$' etc.
unsigned char	0	–	255
byte	0	–	255
int	-32768	-	32767
unsigned int	0	–	65535
word	0	–	65535
long	-2147483648	–	2147483647
unsigned long	0	–	4294967295
float	-3.4028e+38	–	3.4028e+38

double currently same as float
void return type: no return value

Strings

```
char str1[8] =
    {'A', 'r', 'd', 'u', 'i', 'n', 'o', '\0'};
    // Includes \0 null termination
char str2[8] =
    {'A', 'r', 'd', 'u', 'i', 'n', 'o', };
    //Compiler adds null termination
char str3[] = "Arduino";
char str4[8] = "Arduino";
```

Numeric Constants

123 decimal
0b01111011 binary
0173 octal – base 8
0x7B hexadecimal – base 16
123U force unsigned
123L force long
123UL force unsigned long
123.0force floating point
1.23e61.23*10^6 = 1230000

Qualifiers

Static persists between calls
volatilein RAM (nice for ISR)
const read-only
PROGMEMin flash

Arrays

```
byte mypins [] = {2, 4, 8, 3, 6};
int myInts [6];        // Array of 6 ints
myInts[0] = 42;        // Assigning first
                       // Index of myInts
myInts[6] = 12;        // ERROR! Indexes
                       // are 0 though 5
```

Built – in Functions

Pin Input / Output
Digital I/O -- pins 0 – 13 A0 – A5
PinMode(pin,
{ INPUT | OUTPUT | INOUT_PULLUP})
int **digitalRead**(pin)
digitalWrite(pin, {HIGH | LOW})

Analog In – pins A0 – A5
Int **analogRead** (pin)
analogReference(
{DEFAULT | INTERNAL | EXTERNAL})

PWM Out – pins 3 5 6 9 10 11
analogWrite(pin, value) // 0-255

Advanced I/O
tone(pint, freq_Hz, [duration_msec])
noTone(pin)
shiftOut(dataPin, clockPin,
{MSBFIRST | LSBFIRST}, value)
shiftIn(dataPin, clockPin,
{MSBFIRST | LSBFIRST})
unsigned long pulseIn(pin,
{HIGH | LOW}, [timeout_usec])

Time
unsigned long millis()
// overflows at 50 days
unsigned long micros()
// overflow 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)
constain(x, minval, maxval)
map(val, fromL, fromH, toL, toH)

Random Numbers
randomSeed(seed) // long or int
long random(max) // 0 to max-1
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(val)**byte**(val)
int(val) **word**(val)
long(val) **float**(val)

External Interrupts
attachInterrrupt(interrupt, func,
{LOW | CHANGE | RISING | FALLING})
detachInterrupt(interrupt)
interrupts()
noInterrupts()

Libraries

serial – comm.with pc or via RX/TX
begin(long speed) // up to 115200
end()
int **available()** // #bytes available
int **read()**// -1 if none available
int **peek()** // Read w/o removing
flush()
print(data)println(data)
write(byte) write(char*string)
write(byte*data, size)
serialEvent() // called if data rdy

softwareSerial.h – comm. on any pin
softwareSerial (rxpin, txpin)
begin(long speed) // up to 115200
listen() // only 1 can listen
isListening() // at aitme.
read, peek, print, println, write
// Equivalent to serial library

EEPROM.h -- access non-voLatile memory
byte read(addr)
write(addr, byte)
EEPROM[index] // Access as array

Servo.h – control servo motors
attach(pin, [min_usec, max_usec])
write(angle)// 0 to 180
writeMicroseconds(us)
// 1000-2000; 1500 is midpoint
int read()// 0 to 180
bool attached()
detach()

wire.h – I C communication
begin() // Join a master
begin(addr) // Join a salve @ addr
requestFrom(address, count)
beginTransmission(addr) // step 1
send(byte) // step 2
send(char*string)
send(byte*data, size)
endTransmission() // step 3
int avaialable() // #bytes available
byte receiver() // get next byte
onReceive(handler)
onRequest(handler)