**B31DG-EMBEDDED SOFTWARE**

**ASSIGNMENT- 1.**

**GITHUB REPOSITORY:** [GautamiAlagarsamy/Embedded-Software: Assignment-Embedded Software (github.com)](https://github.com/GautamiAlagarsamy/Embedded-Software)

**TASK-1**

This program generates two waveform signals, **Signal A** and **Signal B**.

The waveform consists of repeated cycles, and **signal** **B** should be 50 microseconds.

There are two switch inputs to generate the signals, **Switch1** and **Switch2**, and four parameters, **a**, **b**, **c**, and **d**.

* If **Switch1** is 0, it enables the stream of pulses.
* If **Switch1** is 1, it disables the stream of pulses.
* Similarly, if **Switch2** is 0, the program runs in normal mode,
* If **Switch2** is 1, it runs with a new waveform cycle.

The new waveform cycle parameters are defined as follows:

* **a** is the width of the first pulse, the 2nd pulse is **a + 50 microseconds**, the 3rd pulse is **a + 100 microseconds**, and so on.
* **b** is the width of the space between pulses (all spaces are the same).
* **c** is the number of pulses in a block.
* **d** is the space between pulse blocks.

**TASK 2**

* Write a C++ program for the following question.
* Each student will have a set of (a, b, c, d, and mode) parameters. Calculate your set of parameters using the following information.  
    
  The alphabet is numbered -  
  a=1, b=2, c=3, ... m=13  
  n=13, o=12, p=11, ... z=1
* Using the first 5 letters of your surname (repeat the last letter if less than 5) the parameters are  
  calculated as follows :  
    
  Parameter Example (James Herd)  
  a = first letter \* 100μS a = h \* 100μS = 8 \* 100μS = 800μS  
  b = second letter \* 100μS b = e \* 100μS = 5 \* 100μS = 500μS  
  c = third letter + 4 c = r + 4 = 9 + 4 = 13  
  d = fourth letter \* 500μS d = d \* 500μS = 4 \* 500μS = 2.0mS  
  mode = remainder (fifth letter/4) +1 mode = rem(d / 4) +1 = rem(4 / 4) +1 =1  
    
  Therefore ‘James Herd’ would be given the parameter set 800μS, 500μS, 13, 2.0mS and mode  
  1. The “rem” function is the integer division REMAINDER function, i.e. the remainder when  
  the numerator is divided by the denominator. E.g. rem (10/3) = 1.

**C++ PROGRAM**

The program has 4 possible system modes:

1. If the mode is 1, and **Switch2** is 0, the normal waveform is generated. If **Switch2** is 1, 3 pulses are removed from the pulse block (i.e. **c - 3** pulses in the block) until **Switch2** is set back to 0.
2. If the mode is 2, and **Switch2** is 0, the normal waveform is generated. If **Switch2** is 1, the inverted form of the complete **SignalA** waveform is generated (from the largest pulse to the shortest) until **Switch2** is set back to 0.
3. If the mode is 3, and **Switch2** is 0, the normal waveform is generated. If **Switch2** is 1, an extra 3 pulses are inserted into the pulse block (i.e. **c + 3** pulses in the block) until **Switch2** is set back to 0.
4. If the mode is 4, and **Switch2** is 0, the normal waveform is generated. If **Switch2** is 1, **d** and **b** times are halved until **Switch2** is set back to 0.

**PROGRAM**

#include <iostream>

#include <vector>

const int PULSE\_CYCLE\_LENGTH = 50;// initialize the pulse cycle length 50us

struct WaveformParameters {

int a, b, c, d;

};

class Signal {

// use public and private functions for subroutines

public:

signal(int mode, int switch1, int switch2, WaveformParameters params): mode\_(mode), switch1\_(switch1), switch2\_(switch2), params\_(params) {}

std::vector<int> GenerateSignalA() {

std::vector<int> signal; // If the mode is 1, and **Switch2** is 0, the normal waveform is generated.

if (switch1\_ == 0) {

int num\_pulses = params\_.c;

if (switch2\_ == 1) {// If **Switch2** is 1 use switch case operation for various modes

switch (mode\_) {

case 1://mode-1-3 pulses are removed from the pulse block

num\_pulses -=3;

break;

case 2://mode2= Invert signal

num\_pulses -=num\_pulses;

break;

case 3://mode3= extra pulses are inserted into the pulse block

num\_pulses += 3;

break;

case 4:// mode 4=Half d and b time

num\_pulses =num\_pulses/2;

break;

}}

for (int i = 0; i < num\_pulses; ++i) {

signal.push\_back(params\_.a + i \* PULSE\_CYCLE\_LENGTH);

signal.push\_back(params\_.b);

}

signal.push\_back(params\_.d);

}

return signal;

}

private://function call the inputs and parameters from the subroutine

int mode\_, switch1\_, switch2\_;

waveformParameters params\_;

};

int main() {

int mode, switch1, switch2;

waveformParameters params;

std::cin >> mode >> switch1 >> switch2 >> params.a >> params.b >> params.c >>

params.d;

signal signal(mode, switch1, switch2, params);

std::vector<int> signalA = signal.GenerateSignalA();

std::vector<int> signalB = signal.GenerateSignalB();

// Print signalA and signalB

return 0;

}

#include <iostream>

using namespace std;//initialize the input parameters, modes, and switches.

void generateWaveform(int switch1, int switch2, int a, int b, int c, int d, int mode) {

//use the switch statement to get values for switches and modes for generating waveform.

if (switch1 == 0) {//if switch1=0 the stream of pulses is enabled.

cout << "Stream of pulses enabled." << endl;

}

else {//if switch1=0 the stream of pulses is disabled.

cout << "Stream of pulses disabled." << endl;

}//if switch2=0 the same waveform is generated.

if (switch2 == 0) {

cout << "Running in normal mode." << endl;

}

else {//if switch2=0 the new waveform is generated.

cout << "Running in new waveform cycle mode." << endl;

//use switch case statement for generating pulses of Signal A and B waveform

switch (mode) {

case 1:

if (switch2 == 1) {

c = c - 3;

cout << "3 pulses removed from the pulse block." << endl;

}break;

case 2:

if (switch2 == 1) {

cout << "Generating inverted form of complete Signal A waveform." << endl;

}break;

case 3:

if (switch2 == 1) {

c = c + 3;

cout << "3 pulses inserted into the pulse block." << endl;

}break;

case 4:

if (switch2 == 1) {

b = b / 2;d = d / 2;

cout << "Time for b and d halved." << endl;

}break;

default:

cout << "Invalid mode. Running in normal mode." << endl;

break;

}}}

int main() {//initialize the switch inputs and parameters

int switch1, switch2, a, b, c, d, mode;//call the input switch values to print the output modes

cout << "Enter the value of switch1 (0/1): ";

cin >> switch1;

cout << "Enter the value of switch2 (0/1): ";

cin >> switch2;

cout << "Enter the value of a: ";

cin >> a;

cout << "Enter the value of b: ";

cin >> b;

cout << "Enter the value of c: ";

cin >> c;

cout << "Enter the value of d: ";

cin >> d;

cout << "Enter the value of mode (1/2/3/4): ";

cin >> mode;

generateWaveform (switch1, switch2, a, b, c, d, mode);

return 0;

}

#include <iostream>

#include <string>

using namespace std;

//initialize the alphabet in ascending order from a to m

int number\_of\_alphabet(char alphabet) {

if (alphabet >= 'a' && alphabet <= 'm') {

return alphabet - 'a' + 1;//initialize the alphabet in descending order from n to z

} else if (alphabet >= 'n' && alphabet <= 'z') {

return 13 - (alphabet - 'n');

} else if (alphabet >= 'A' && alphabet <= 'M') {

return alphabet - 'A' + 1;

} else {

return 13 - (alphabet - 'N');

}}

int rem(int num, int den) {

return num % den;

}

int main() {//get the first and last name inputs from the console

string first\_name, last\_name;

cout << "Enter your first name: ";

cin >> first\_name;

cout << "Enter your last name: ";

cin >> last\_name;//calculate the parameter values from the first 5 values of the last name

int length = last\_name.length();

if (length < 5) {

last\_name += last\_name.substr(0, 5-length);

}

int a = number\_of\_alphabet(last\_name[0]) \* 100;

int b = number\_of\_alphabet(last\_name[1]) \* 100;

int c = number\_of\_alphabet(last\_name[2]) + 4;

int d = number\_of\_alphabet(last\_name[3]) \* 500;

int mode = rem((last\_name[4] <= 'Z' ? last\_name[4] - 'A' + 1 : last\_name[4] - 'a' + 1), 4) + 1;

//Print the parameters and modes for waveform

cout << "The parameter set for the name " << " is " << a << " μS, " << b << " μS, " << c << ", " << d << " μS, and mode " << mode << "." << endl;

return 0;

}

OUTPUT -RESULT

Shape

Description automatically generatedText

Description automatically generated

**Explanation:**

This program generates two waveform signals, **SignalA** and **SignalB**.

The waveform consists of repeated cycles, and **SignalB** should be 50 microseconds.

There are two switch inputs to generate the signals, **Switch1** and **Switch2**, and four parameters, **a**, **b**, **c**, and **d**.

If **Switch1** is 0, it enables the stream of pulses; if **Switch1** is 1, it disables the stream of pulses. Similarly, if **Switch2** is 0, the program runs in normal mode, and if **Switch2** is 1, it runs with a new waveform cycle.

The new waveform cycle parameters are defined as follows: **a** is the width of the first pulse, the 2nd pulse is **a + 50 microseconds**, the 3rd pulse is **a + 100 microseconds**, and so on. **b** is the width of the space between pulses (all spaces are the same). **c** is the number of pulses in a block, and **d** is the space between pulse blocks.

This is a C++ program that takes the user's first and last name as input, calculates a set of parameters for the name and outputs it to the console.

The program uses the function "number\_of\_alphabet" to calculate the position of an alphabet in the English alphabet (from 1 to 26) and maps it to a number.

The program then concatenates the first 5 characters of the last name and calculates the values of the parameters a, b, c, d and mode by using the first 5 characters of the last name as input. The value of a and b are calculated as the position of the first and second characters of the last name multiplied by 100. The value of c is calculated as the position of the third character of the last name plus 4. The value of d is calculated as the position of the fourth character of the last name multiplied by 500. The value of mode is calculated as the remainder of the position of the fifth character of the last name divided by 4 plus 1.

Finally, the program outputs the values of the parameters to the console.

**FIRST NAME: GAUTAMI**

**LAST NAME: ALAGARSAMY**

**Parameters calculation : A=1,L=12,A=1,G=7 and A=1.**

1. Parameter 'a' calculated as first letter\*100 = 1\*100 = 100=>**a = 100**
2. Parameter 'b' calculated as second letter\*100 = 12\*100 = 1200 => **b = 1200**
3. Parameter 'c' calculated as third letter+4 = 1+4=5=> **c = 5**
4. Parameter 'd' calculated as fourth letter\*500 = 7\*500 = 3500=> **d = 3500**
5. Mode is calculated as rem(fifth letter/4) + 1 = rem(1/4) + 1 = 2. Hence, we use **mode 2.**

**OUTPUT**

**A picture containing text, indoor, electronics, computer

Description automatically generated**

**Diagram

Description automatically generated with medium confidence**

Python Code:ESP32 C3

const int ledPin1=9;// initialize the input pin of LED1

const int ledPin2=10;//initialize the input pin of LED2

const int pushButtonPin1=4;//initialize the input pin of pushbutton switch1

const int pushButtonPin2=5;//initialize the input pin of pushbutton switch2

int  pushButtonState1=0;//assign the state of pushbutton switch1

int  pushButtonState2=0;//assign the state of pushbutton switch2

//Last Name:ALAGARSAMY  First five alphabets of surname A=1,L=12,A=1,G=7, A=1

int a = 100;// parameter 'a' calculated as first letter\*100 = 1\*100 = 100

int inv = 4900;// parameter used in mode '2' where we reverse the signal, hence highest value is a+(c-1)\*b 100+(5-1)\*1200=4900

int b = 1200;// parameter 'b' calculated as second letter\*100 = 12\*100 = 1200

int c =5; // parameter 'c' calculated as third letter+4 = 1+4=5

int d = 3500;// parameter 'd' calculated as fourth letter\*500 = 7\*500 = 3500

// mode is calculated as rem(fifth letter/4) + 1 = rem(1/4) + 1 = 2. Hence we use mode 2

void setup()

{

  pinMode (pushButtonPin1, INPUT );

  pinMode (pushButtonPin2, INPUT );

  pinMode (ledPin1,OUTPUT);

  pinMode (ledPin2,OUTPUT);

 }

 void loop() {

 pushButtonState1=digitalRead(pushButtonPin1);

 pushButtonState2=digitalRead(pushButtonPin2);

 if(pushButtonPin1== HIGH)//condition if switch1 is high the LEDs are in OFF condition

 {

   digitalWrite(ledPin1, LOW);

   digitalWrite(ledPin2, LOW);

   }

   else

   {

   if(pushButtonPin2 == LOW)//condition if switch2 is low the output waveform is normal.

   {

     digitalWrite(ledPin2, HIGH);

     delay(50);

     digitalWrite(ledPin2, LOW);

     }

     if(pushButtonPin2 == HIGH)//condition if switch2 is high the new output waveform is generated.

     {

      digitalWrite(ledPin1, HIGH);

      delay(a);// parameter a=width of the first pulse

      c=c-3;

      digitalWrite(ledPin1, LOW);

      delay(b);

      a = a+50;// the 2nd pulse b is a + 50 microseconds

      digitalWrite(ledPin1, LOW);

      delay(d);

      }

      else

      {

        digitalWrite(ledPin2, HIGH);

        delay(50);

        digitalWrite(ledPin2, LOW);

        for(int i =0; i<c;i++)// c is the number of pulses in a block

        {

          digitalWrite(ledPin1, HIGH);

          digitalWrite(ledPin2, LOW);

          delay(inv);

          digitalWrite(ledPin1, LOW);

          delay(b);

          inv = inv-50;

          digitalWrite(ledPin2, LOW);

          }

          digitalWrite(ledPin1, LOW);

          delay(d); d is the space between pulse blocks.

          d=d/2;b=b/2;

          digitalWrite(ledPin2, LOW);

          }

          }

          }

**OUTPUT DISCUSSION:**

The connections are given in the breadboard with a push button switch to pin 4 and push button 2 to pin 6 of the ESP32 C3 board. The LED1 and LED2 connections are given to Pin 1 and Pin3 respectively. The supply and ground connections are given as per the circuit diagram. The Port and boards are selected, and the program is dumped into the ESP32 C3 board. The program is compiled and debugged to get the output signal.

The push buttons are pressed to generate the stream of pulses on both LEDs as per commands.

The new waveform cycle parameters are defined as follows: a is the width of the first pulse, the 2nd pulse is a + 50 microseconds, the 3rd pulse is a + 100 microseconds, and so on. b is the width of the space between pulses (all spaces are the same). c is the number of pulses in a block, and d is the space between pulse blocks.

To calculate the parameters and mode as per task 2 the following formulae are used.