Alex Fuhr, Daniel Glauser

Software Design Project Documentation

1. Introduction and Purpose

The main goal of the project was to create a working Proteus program which could both receive IR signals and display the correct values for the signal’s frequency. The program was required to wait for a button press to begin, receive and promptly display the value of one out of the five frequencies of the signal on the Proteus screen without displaying intermittent values, and close with a second button press.

1. Planning

Brainstorming Concepts:

* Measure the interval of the on and off signals and convert both into a frequency. If they match and fall into one of the available frequencies, display the frequency.
* Store an array of Boolean values over one period of data. Use the length of array as the period in milliseconds.
* Collect one second of Boolean data. Using this, find how many periods occurred (the frequency).
* Use the first concept, but instead of recording start and end times, increment a time variable by 1 ms (the sleep time) within each loop cycle.
* Record 2 or 3 period data sets at a time and check if they all give the same frequency.

Pseudocode:

Declare a variable for the frequency and set it to 0

Define constants PERIODS and TIMEOUT

Initialize input components

Clear the LCD and display a welcome message

Wait for middle button press

Do

Clear the LCD

Display the frequency in Hz on the LCD

Declare variables for the time and timeout intervals and set both to 0

While (off signal)

Sleep (1)

Increment timeout interval by one

If timeoutInterval > TIMEOUT

Set frequency to 0 Hz

Continue to next loop iteration

Loop PERIODS times

While (on signal)

Sleep (1)

Increment time by 1

While (off signal)

Sleep (1)

Increment time and timeout by 1

If timeoutInterval > TIMEOUT

Set frequency to 0 Hz

Continue to next loop iteration

Calculate on frequency from the time as 1000 \* PERIODS / time

Set frequency to one of the given values if it is within 3 Hz of it.

While middle button is not pressed

Display a closing message

1. Developer Description

The program began with the standard library declarations of FEHLCD.h, FEHIO.h, and FEHUtility.h. Then the function determineFrequency(int) was declared using a function prototype, and the return type was designated as an integer. The integer variables for period length, pause length, and signal frequency w declared. The input to the Proteus board was declared through the use of ButtonBoard and DigitalInputPin objects. The LCD on the board was cleared, and the font was formatted. A message was output to the screen prompting the user to press the middle button on the button board. When pressed, integer variables for pause length, and time were declared. A predetermined number of wave periods was measured, incrementing time by intervals of 1 ms throughout this duration. The measured frequency was calculated from this time and inserted as the parameter of the determineFrequency(int) function. In this function, if the value is within the allotted range of an allotted frequency, then the function returns the frequency value, or it returns -1. IF the result was not -1, the frequency variable was set to the result, and displayed to the screen. This process is repeated until the middle button is pressed again.

Variables and their uses:

* PERIODS – Constant representing the number of periods to record data for.
* TIMEOUT – Constant representing the number of milliseconds of off data that should occur before a timeout condition is triggered.
* frequency – Frequency displayed to the LCD display. Either one of the given frequencies or 0 for no signal.
* buttons – Proteus button board, used to check for pressing of the middle button.
* ir – Proteus IR receiver, used to check for an “on” or “off” signal to trigger each loop
* timeoutInterval – Milliseconds recorded during an “off “ signal loop, used to determine if timeout has occurred.
* time – Time in milliseconds recorded for the set number of periods, used to calculate measuredFrequency.
* timeout – Used to determine if timeout has been reached. True if an off loop has ran longer than TIMEOUT milliseconds.
* measuredFrequency – The raw frequency as calculated from the IR signal time.
* setFrequency – The frequency either matched by determineFrequency(), or -1 for no match.
* possibleFrequencies – The array of frequencies the IR beacon can output, used in determineFrequency() to match to the measured frequency.

Functions:

* int main(void) – Main function of program. Displays beginning message, waits for middle button press, samples frequencies, and then displays closing message when the middle button is pressed again. Returns 0 for a successful completion.
* int determineFrequency (int measuredFrequency) – Attempts to match each of the possible frequencies to the input frequency. Returns one of these frequencies if the measured frequency is within 3 Hz of it. If no match is found, returns -1. Example call: int setFrequency = determineFrequency(measuredFrequency)

Assumptions:

* The acceptable range of error is ± 3 Hz.
* The button press to turn off the program is held long enough to occur during the end of the loop.
* The frequency given is either 25 Hz, 40 Hz, 65 Hz, 80 Hz, or 110 Hz.
* A consistent “on” signal will never be given, only a consistent “off” signal when no IR wave is being transmitted.

Performance/Limitations:

* The program requires at least the number of periods specified by PERIODS to obtain a result from a given frequency.
* The program requires one second to determine that there is no frequency.
* The program can only detect 25 Hz, 40 Hz, 65 Hz, 80 Hz, or 110 Hz.
* The program cannot run again after the middle button is pressed the second time.

1. User Description

The program began with the declaration of variables to test the frequency. The user is prompted to press the middle button, and the conditional statements are executed. The program records the length of time for five periods of the IR signal. If the calculated frequency is within 3 Hz of an allotted frequency, the program sets that frequency, and displays the frequency on the output screen until the middle button is pressed again, or the signal frequency is changed.

In order to run the program on the Proteus device, place a button board input in the rightmost group of input pins, and the IR receiver in the leftmost column of input pins. To start the program, turn on the Proteus board by pressing the button on the right hand side on the top of the board. When prompted to press the middle button to start the program, press the middle button of the button board attachment, and the Proteus starts to output results. If an IR signal is not within the range of the receiver, the board displays a frequency of 0 Hz. Otherwise it displays a frequency of 25, 40, 65, 80, or 110 Hz. To stop the reading of frequencies, press the middle button on the button board.

1. Problems During Testing

The first version of the program which measured frequency from 1 period of on and off values separately was unable to display one of the possible frequencies. This occurred because a delay occurred between loops that caused the measurement of on time to be shorter than expected. This was fixed by recording data for multiple periods rather than just one. Another problem during the testing of the program was the Proteus constantly displayed a frequency of 40 Hz for receiving a signal of 65 Hz. The error was fixed, when it was realized that the program was testing for 55 Hz rather than 65 Hz.