Development of Auto-Generated Code for Programming PIC16F877A Microcontroller for Microprocessor System Subject

Hilario A. Calinao Jr., MS-ECE
Electronics Engineering Department
Bulacan State University
Malolos City, Philippines
hcalinao@gmail.com

***Abstract* - This study aimed to develop system that can be used by the student on building a microcontroller unit based project utilizing the PIC16F877A microcontroller and for students having Microprocessor System subject. The study utilized the combination of textual programming and visual programming technique. The AGCPIC system is composed of software and hardware. The software is used to create schematic diagram and flowchart. The software generates source code that is equivalent to the configuration and system flowchart created by the user.**

KEYWORDS: AGCPIC – Auto-Generated Code for Programming PIC16F877A Microcontroller

I. INTRODUCTION

One of the challenging subjects of Electronics Engineering students is the Microprocessor System. The Microprocessor System subject covers topic on microcontroller system, in which student should learn the engineering concept that will help them in developing an embedded system.

The subject covers concepts involving microprocessor and microcontroller systems architecture or organization which includes microprocessor and microcontroller programming, interfacing with input and output devices, data transfer between microcontrollers based circuit and personal computer via serial or parallel port and memory systems. The technical knowledge that the students gained from this subject becomes necessary in the development of an embedded system project or a microcontroller based project.

Students agreed that the most difficult phase in developing or building an embedded system is writing codes or programming. Testing the project is the second of the difficult phase that the students encountered. Although there are simulation program to test the source code, it does not produced a real time response. The actual response of the source code for microcontroller must be observed using the actual hardware. Interfacing the external components and modules placed third of the difficult phase that the students encountered.

Other concerns that the student should address is flowcharting. The knowledge of the student on flowcharting is also important and it helps to understand the programming of an embedded system. The program flowchart is a graphical representation of the required functionality of the software. Creating a flowchart is important before programming because it guides the programmer in visualizing the flow of the program. Without the knowledge in flowchart, the system will possibly result to unorganized routines and flow of the whole program.

This study would like to present a visual programming technique on programming the microcontroller for building an embedded system project. The visual programming and interactive software is used to guide and simplify the microcontroller programming. This project possibly makes an advantage to the students and enhances the knowledge that is necessary for the development of an embedded system projects.

II. VISUAL PROGRAMMING LANGUAGE

Visual Programming Language. According to Loredana (2010), the visual programming language allows the use of icons which is an object with dual representation of a logical part (meaning) and a physical part (image). Dr. Loredana also states that “the visual programming language has three construction rules that are used to arrange icons: horizontal concatenation, vertical concatenation and spatial intersection” (p. 9).

The study also contains three construction rules and this is used for an icons, blocks and lines that represent the visual flow and routines of the program. Each block has an equivalent textual code that is arrange in accordance to its symbol. The AGCPIC software also uses library that contains the necessary information to produce the required operation. The AGCPIC library contains all the necessary codes for every function needed during programming.

A. *Interactive approach in Visual Programming*

According to Gorgan (1999), faculty member of Computer Science in Technical University of Cluj-Napoca, the important feature of the interactive application is the user control, where developed using graphical user interface. Visual programming should provide an interactive application that consists of "conceptual component" and "interactive component." Conceptual component involves functional and application. Visual programming technique should assist the user control that is a user oriented interface. The flow of data can directly manipulate through graphical interface represented by wires and arrows. The development of visual programming software must have better user control interface so that a person as programmer can use the program easily. **V**isual programming uses visual notations such as line, geometric shapes, patterns and symbols resembles the non-textual part of the visual programming language.

On this study, the components, parts and modules will be programmed based on interactive approach in configurations and interfacing. The creation of program flow will also base on interactive approach that uses visual notations such as lines, geometric shapes, patterns and symbols to represent the processes, routine and program flow.

B. *Productivity Improvement through Visual Programming.*

According to Baroth and Hartsough (1994), "the productivity improvement of software development is due to improved communications to the costumer, developer and computer or hardware" (p. 10). The visual programming tools usually used to simplify and improve the development of a system that needs programming. Comparing to text-based programming, this visual programming improves productivity of such system. The experience report of Baroth and Hartsough (1994) on visual programming proved that visual programming provides productivity improvement. By comparing the visual programming to conventional text-based programming the study shows that the improvements are from four to ten times.

This study makes use of visual programming to develop an embedded system project. The customer and developer may be same person as student and the computer as the AGCPIC system. With good communications between customers, the developer and the computer productivity improvement will improve. Also, the combinations visual programming and interactive approach, the study is assumes to improve the productivity of building an embedded system.

III. PROGRAMMING LANGUAGE USED AS PLATFORM FOR AGCPIC SOFTWARE

The programming language used by the researcher as platform to develop the auto-generated code for programming PIC16F877A software is the Microsoft Visual Basic 2010 of Microsoft Corporation. The Microsoft Visual Basic 2010 is included on Microsoft Visual Studio 2010 Ultimate version 10.0.30219.1 package which contains different development tools that are useful in the development of Windows application software. VB10 included functions, commands, buttons and other tools necessary for the development of a software. VB10 has also a database handling capability like its local database feature that can handles information, properties, images and commands. The data and information can store and retrieve from the local database of visual basic 2010 software. The visual notations, lines, graphs, images and properties of the proposed software can simply be used using visual basic 2010. This programming language is ideal for the development of the proposed project.

IV. PROGRAMMING LANGUAGE FOR PIC16F877A MICROCONTROLLER SOURCE CODE

The programming language used to create a source code for the embedded system project is BASIC language or the Beginner's All-purpose Symbolic Instruction Code. The BASIC language for microcontroller is easy to understand language that can be easily used by the student. Unlike C-programming language, the BASIC language does not used of semicolon (;) at the end of each instruction lines which can possibly cause syntax error when omitted. One of the software that utilized the use of BASIC language is the MikroBASIC Pro for PIC version 6. The researcher will use the programming software to build the hexadecimal language or the machine language for PIC16F877A microcontroller. This machine language will be downloaded to the microcontroller using the hardware programmer software. The hardware programmer software is separate part of the system that is used only to load the source code into the PIC16F877A microcontroller.

A. Developing the Product

The development of the auto-generated code for programming the PIC16F877A microcontroller need to be carefully examines. The programming tools and other software that can be used to develop the interactive system and used for representation of each function will be considered. Programming codes necessary for each module must be carefully check and validate if it can be integrated on the software using Visual Basic language. Among the Visual Basic family developed by Microsoft, the researcher will use the VB10 or the Visual Basic 10. The hierarchical structure of the AGCPIC software is shown in Fig. 1.

The output of the software will be used as source code of the project. The source code will be downloaded to an embedded system that uses PIC16F877A microcontroller. The language that will use by the researcher for the PIC16F877A is BASIC language for microcontroller. The symbols, dataflow and property created by the user will be converted into PICBASIC codes in a form of text document.

The source code generated by the software will be uploaded to the testing hardware. The hardware will be called the development board.

The development board will use three different sources of power. The battery terminal, the USB terminal and the DC source will be available to give power to the whole development board. For the input devices, there will be an 8-bit switches, keypad matrix

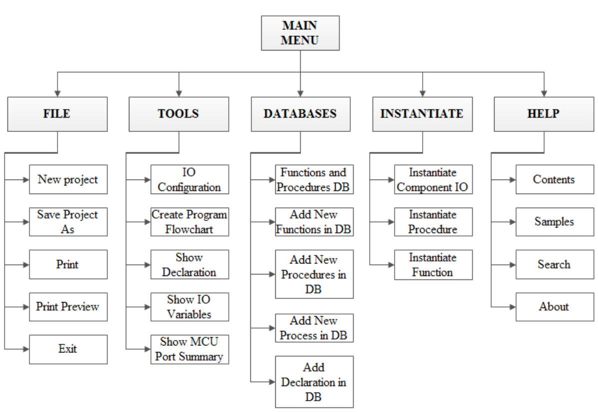


Figure 1. Hierarchical Structure of the AGCPIC Software

and analog voltage source. For the output devices, there will be an LCD display, 5x7 LED matrix, two 7-segment display, 8-bit LED and speaker is use to indicate the output response of the program. Fig. 2 shows the block diagram of the AGCPIC system development board that can be used for testing the validity of the generated code.

There are available IO-ports for developing some projects that uses PIC16F877A microcontroller. The ports available are PORTA (6 terminals), PORTB (8-terminals), PORTC (8-terminals), PORTD (8-terminals), PORTE (3-terminal), SPI (Serial Peripheral Interface) terminals, Servo motor terminals and UART terminals. The main component of the whole development board will be the PIC16F877A microcontroller. The microcontroller is the main processing unit which controls the whole development board.

V. INPUT AND OUTPUT DEVICES THAT CAN BE PROGRAMMED USING AGCPIC SOFTWARE

The list of input devices that can be used to program the microcontroller are 8-bit switches, 4x4 keypad matrix and two analog voltage source. The researcher used push button switch for 8-bit switches. Each switch is based on push-to-make functionality which means that the user needs to push the button in order to have continuity on the circuit. The 4x4 keypad matrix is one of the inputs that are included on the proposed project. This keypad is commonly used for embedded system that requires numeric inputs to the microcontroller. The two analog voltage sources are used to simulate different sensors that delivers analog signal to the microcontroller.

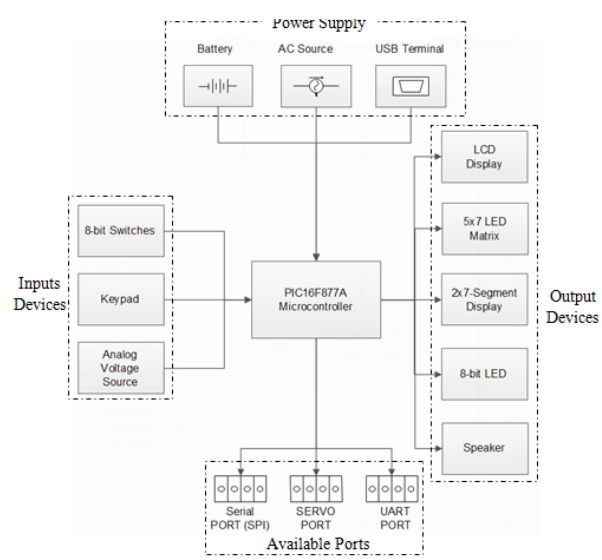


Figure 2. Block Diagram of AGCPIC System Development Board

The proposed project is composed of the following output devices; the 8-bit LED, 5x7 LED Matrix, two 7-Segment display, LCD screen display and Speaker. The researcher used to organize other terminals based on the required function of the user. These includes Serial communication port terminal, servo motor terminal, In-Circuit Serial Programming terminal, PortA, PortB, PortC, PortD and PortE terminals. All of these terminals are commonly used in embedded system project.

VI. BASIC SCHEMATIC DIAGRAMS AVAILABLE FOR AN EMBEDDED SYSTEM USING AGCPIC SOFTWARE

The available schematic diagram that can be programmed using the AGCPIC software that is used to visualize the hardware representation of the project is shown in Fig. 3.

The user can select and create any of the schematic diagrams for different input-output module together with the microcontroller that is needed to represent the project. The available schematic diagram has an equivalent hardware part on the Hardware Simulation Board. The available schematics for input are 8-bit switches, 4x4 keypad matrix and two analog voltage source. The available schematics for output are 8-bit LED, two digit seven-segment display, 5x7 LED matrix, LCD- liquid crystal display and speaker. The main controller which is the PIC16F877A microcontroller is also given.

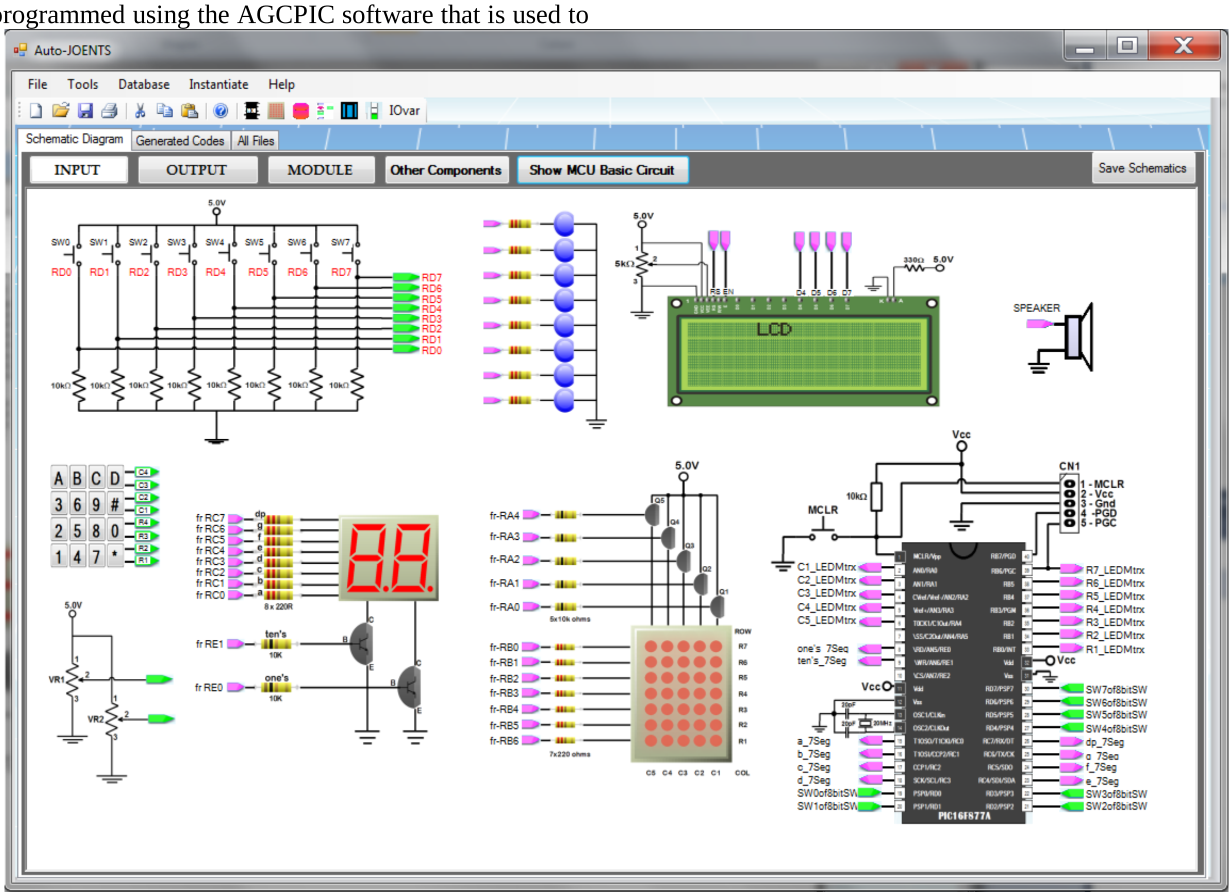


Figure 3. Available Schematic Diagrams Using the AGCPIC Software

VII. HARDWARE USED FOR TESTING THE CODES GENERATED BY THE SOFTWARE

The hardware part of the proposed project is composed of the Main Module and the I/O Module. Fig. 4 shows the Development board schematic diagram for Main Module of the system. This includes the PIC16F877A microcontroller that serves as the main control unit of the system. This schematic diagram can be used as reference on where to connect the schematics made using the AGCPIC software to the development board of the system.

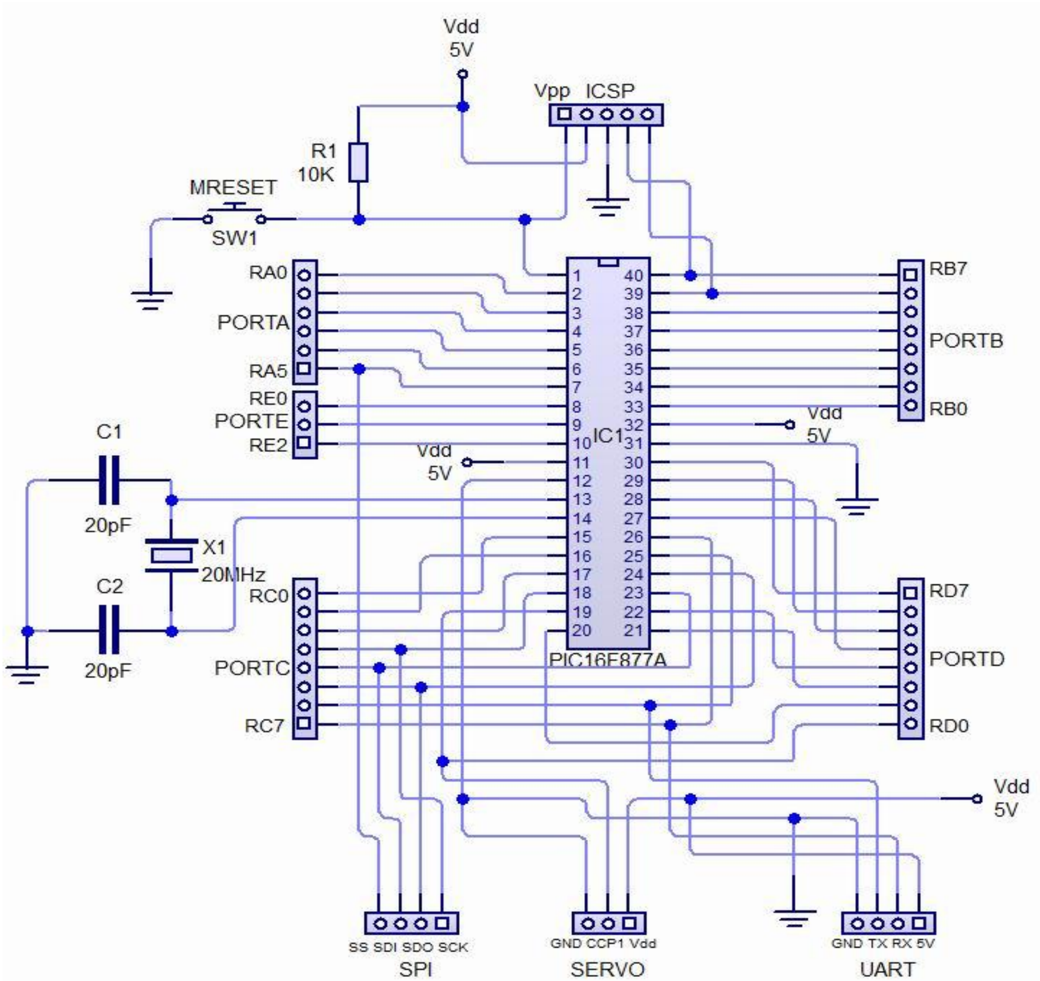


Figure 4. Schematic Diagram of the Development Board for Main Module

The project includes the readily available input and output components that can be used for experimentation and testing of the source code that was made by the user. There are two available analog voltage sources. Fig. 5 shows the schematic diagram of the analog voltage source used on the system. VR1 and VR2 are potentiometers that are used to vary the amount of voltage which can be connected to AN2 and AN3 terminals of the PIC16F877A microcontroller, respectively. The two potentiometers can output a voltage ranging from 0 to 5 volts. The analog voltage from the potentiometers can be used to simulate the analog signal from different sensors and other components that gives an analog signal.

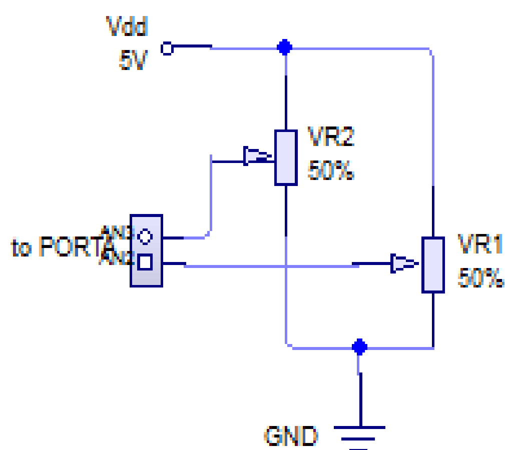


Figure 5. Analog Voltage Source Schematic Diagram

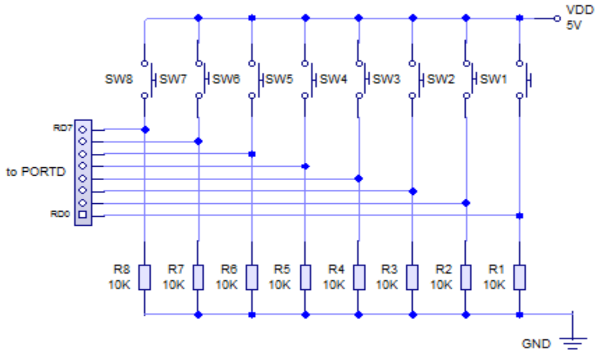


Figure 6. Schematic Diagram of the 8-Bit Push Button Switches

Other input component available on the project is the 8-bit push button switches as shown in Fig. 6. The schematic diagram of 8-bit push button switches on Fig. 6, creates an 8-bit data entry that can be used by the PIC16F877A microcontroller as an input signal. The circuit diagram is composed of 8 push button switches connected from the supply to Port D terminal of the microcontroller.

The project has several components that serve as an output indicator of the processed data. Fig. 7 shows schematic diagram of an 8-bit LED indicator. The LED's are connected from Port B of the PIC16F877A microcontroller in series with the resistor connected to ground.

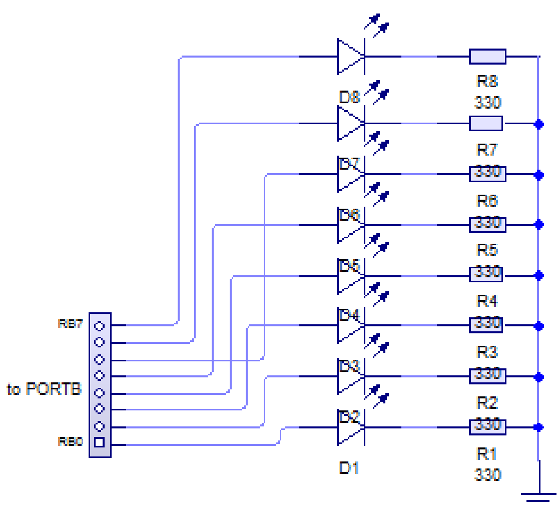


Figure 7. 8-Bit LED Indicator Schematic Diagram

Fig. 8 shows the two-digit 7-segment display schematic diagram which is another output data indicator. The output data from Port B can be displayed in numerical equivalent using 2 digit seven-segment display.

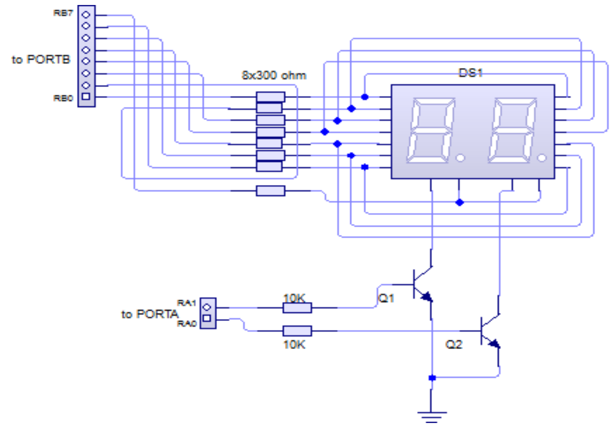


Figure 8. Two-Digit 7-Segment Display Schematic Diagram

Fig. 9 shows a 5x7 LED matrix schematic diagram which can be used to display an alpha-numeric character. The LED matrix shown in Fig. 9 is a common-row cathode. The LED matrix is composed of five columns which are attached to the emitter terminal of the transistors and seven rows which are connected to Port B via series resistors.

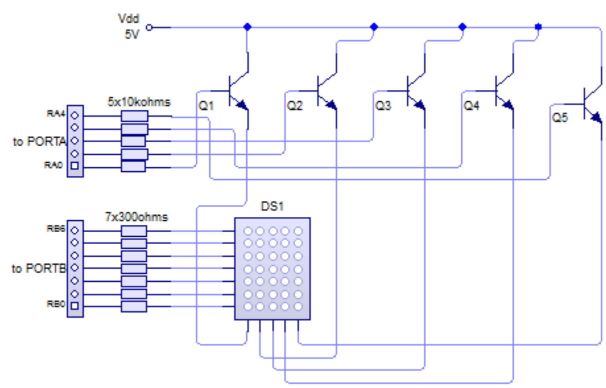


Figure 9. 5x7 LED Matrix Schematic Diagram

Fig. 10 shows a Liquid Crystal Display (LCD) schematic diagram which are used on the project to display alpha-numeric characters, words, processed data and even statements necessary to inform the user. The data terminals of the LCD, D4 to D7, are connected to RB4 to RB7 of the microcontroller. The control terminals of the LCD display, E and RS, are connected to RB2 and RB3 of the microcontroller, respectively. The A terminal is connected to 5 volts supply and K terminal is connected to ground.

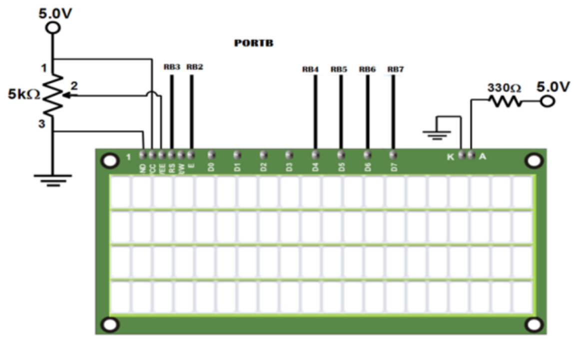


Figure 10. LCD Display Schematic Diagram

A. Development board PCB layout

The hardware that can be used to test and simulate the generated source code using the AGCPIC software is the Development Board for PIC16F877A. The Development board for PIC16F877A was developed by the researcher in order to test the source code. The development board was specially designed board that is based from the AGCPIC software. The input and output devices on the software have an equivalent hardware on the development board.

The development board is composed of two major parts; the Main Module and the I/O Module or the Input-Output Module. The two main parts can be used separately which means that the Main Module can operate separately without the I/O Module. Likewise, the I/O Module can be used separately without the Main module. Fig. 11 shows the PCB layout of the development board for main module. The PCB needed for this project is a double-sided PCB to reduce the complexity of the PCB layout.

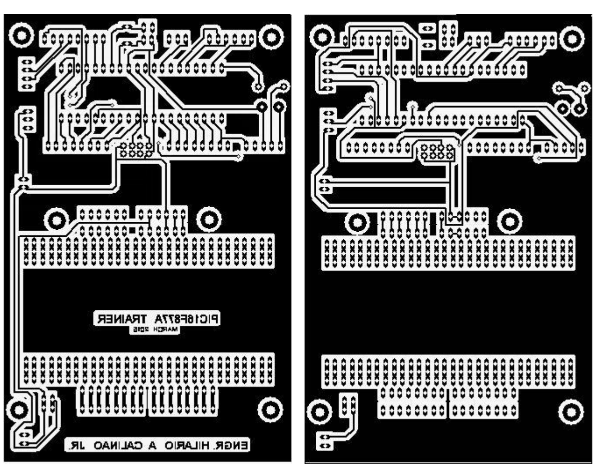


Figure 11. Development Board's PCB Layout for Main Module

The input-output PCB layout that is used for testing and validation of the generated source code for this project contains most of the input devices and output devices. Fig. 12 shows the Input-Output Board or I/O Board layout of the project.

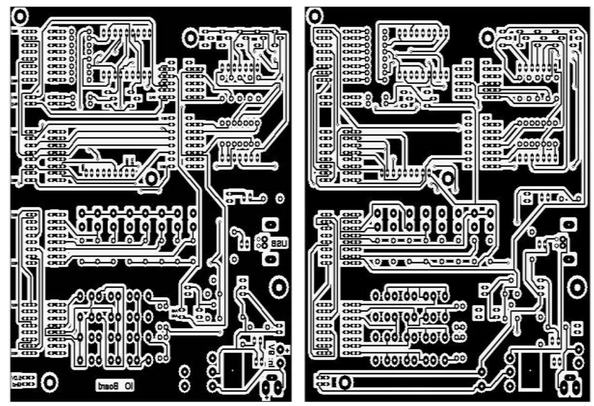


Figure 12. Development Board's PCB Layout for Input-Output Module

B. Development board parts placement.

Fig. 13 shows the main board component placement plan for the hardware part of the subject. On the component placement shown in Fig. 13, the main component of the project is placed, which is the PIC16F877A microcontroller. The main board has breadboard which is ideal for prototyping and development of other projects. The main board can be operated independently provided that the supply voltage is equivalent to regulated 5 volts. The supply must be connected on any 5.0V terminal and ground terminal. All of the available ports are also on the main module so that other types of devices can be configure and connected to the main module without utilizing the I/O module of the project. The researcher positioned a small type of breadboard in order for the user to put a customized circuit unto it. All of the slots on the breadboard are internally connected on the header pins located on its side.

The I/O Module is composed of the different input components and the construction of the input components or devices has default connection to the designated port terminals. The analog voltage source with default connection to Port A, the 4x4 keypad matrix with default connection to Port D and the 8-bit push button switch with default connection to Port D.

Even though the devices have default connections, the user can select other terminals if the user wants to select other available port terminal. In order change the default connections, the user need to switch-off the interface switch for the selected devices. Every input-output device has corresponding interface switch that are user-oriented.

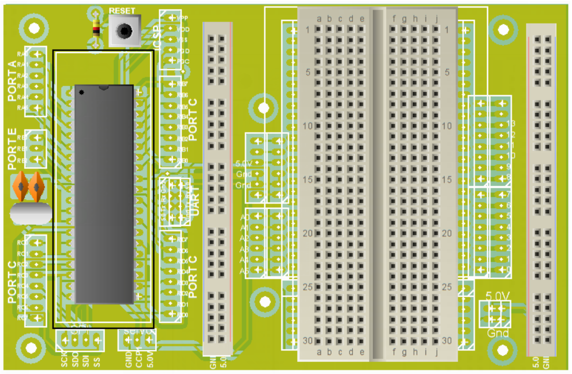


Figure 13.Development Board's Component Placement for Main Module

The output components of the I/O Module is composed of the following parts; the speaker, the 8-bit LED with an default connection to Port B, the 2-digit seven-segment display with an default connection to Port B (data) and Port A (control signal) and the 5x7 LED matrix with an default connection to Port B (row) and Port A (column). The other components that can be seen on the input-output module are used for power supply and regulator circuit. Fig. 14 shows the Input-Output component placement.

VIII. STEPS IN CREATING THE SOURCE CODE FOR AN EMBEDDED SYSTEM

According to Alciatore (2011), the methods to build an embedded system includes the following: 1) Problem definition; 2) Selection of microcontroller type; 3) Identification of required circuits; 4) Selection of programming language to be used; 5) Drawing of schematic diagram; 6) Drawing of the program flowchart; 7) Writing of source codes for the project; 8) Testing of the project.

The steps developed in this study is related to the steps defined by Alciatore. Through the development of this study, the basic programming steps are: 1) Problem definition; 2) Creation of schematic diagram; 3) Creation of flowchart; and 4) Project testing.

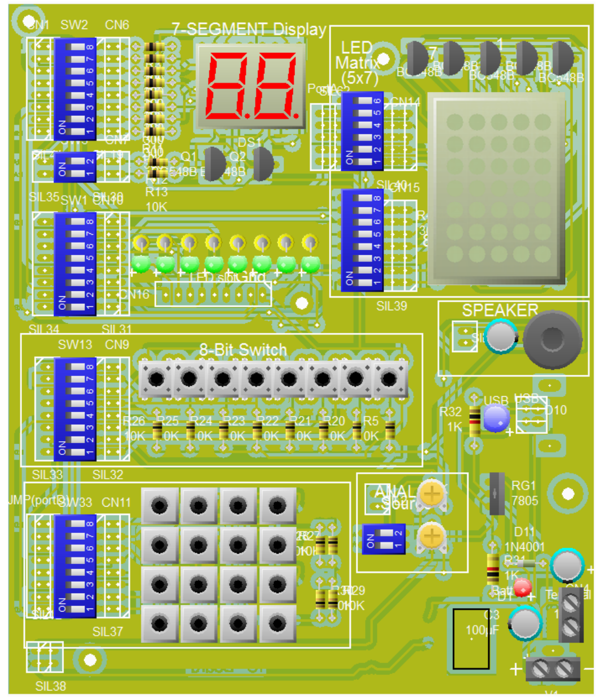


Figure 14. Development Board Component Placements for Input-Output Module

Fig. 15 shows that actual project development board which shows the main module and input-output module.

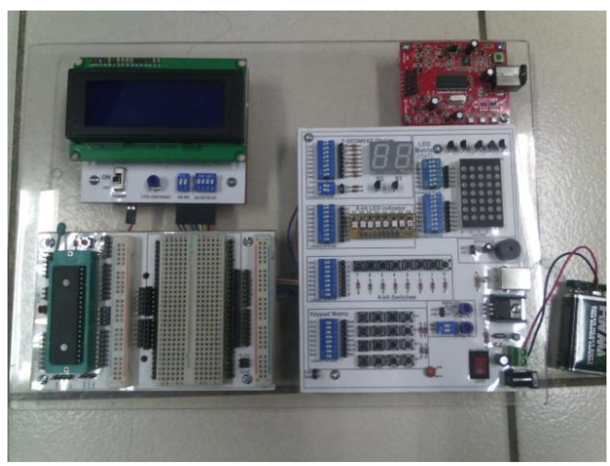


Figure 15. Actual Development Board of the Auto-Generated Code For Programming PIC16F877A Microcontroller

IX. RELIABILITY TESTING FOR AGCPIC SYSTEM

The ISO 9126, the information technology - software product quality defines reliability as the ability of the software to maintain the particular performance level when it is used for a certain conditions. The documentation also states that the software is not a product that worn-out or depleted. The reliability is maintained when the specific condition based on the requirements, design and the way to implement the software’s are met. And to minimize faults during implementation, the software must be tested. Through certain levels of testing, the reliability of the system can be determined. To determine if the system developed on this study is reliable, this study used unit testing, integration testing and systems testing.

A. Unit testing

Table 1 summarizes the unit testing results on the hardware Input parts that was conducted by the researcher on the system. The first column of Table 1 is the functions that need to be performed and the second column is the part of the hardware that performs the specified function. Third column of Table 1 shows the results of the testing on the input parts of the actual module.

Table I. Unit Testing Results on the Hardware Input Parts

Function	PART/S	Result
Send High Signal to Microcontroller	SW0	Passed
	SW1	Passed
	SW2	Passed
	SW3	Passed
	SW4	Passed
	SW5	Passed
	SW6	Passed
	SW7	Passed
Respond to Pressed Key	Keypad Matrix	Passed
Generates 0-5Volts	Analog source 1 (VR1)	Passed
Generates 0-5Volts	Analog source 2 (VR2)	Passed

Table 2 shows the result of the unit testing on the Output parts of the system that was conducted. The first column is the function and the second column is the output part of the system that needs to perform the specified function and the third is the result of the actual testing.

Table II. Unit Testing Result on the Hardware Output Parts

Functions	PART/S	Result
Blinking	LED0	Passed
Blinking	LED1	Passed
Blinking	LED2	Passed
Blinking	LED3	Passed
Blinking	LED4	Passed
Blinking	LED5	Passed
Blinking	LED6	Passed
Blinking	LED7	Passed
Blinking	LED0-7	Passed
Running Light (Right to Left)	LED0-7	Passed
Running Light (Left to Right)	LED0-7	Passed
Display numbers	7-Segment One's	Passed
Display numbers	7-Segment Ten's	Passed
Counting	7-Segment One's and Ten's	Passed
Display numbers	LED Matrix	Passed
Display Alphabet	LED Matrix	Passed
Counting	LED Matrix	Passed
Display numbers	LCD Display	Passed
Display Alphabet	LCD Display	Passed
Generate Sound	SPEAKER	Passed

B. Integration Testing

The integration testing was conducted by the researcher to check the compatibility issues between main module and input-output module. The integration testing was done after each parts, components and modules was placed. The integrated hardware that is composed by main module and input-output module is called the hardware development board.

During integration testing, the main module was connected to the input-output module using header wires. After hard-wiring, the researcher downloaded a sample program to test the communication and functionality of each part of the AGCPIC system. Table 3 shows the summary of integration testing conducted by the researcher. The signal source is connected to specific input-output part and test the communications function of each input-output part.

Table III. Summary of Integration Testing Result

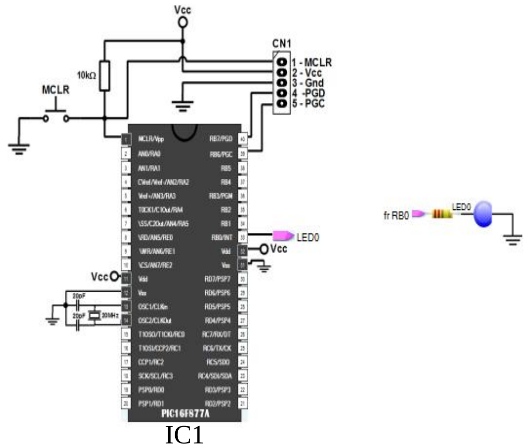
Signal Source	Function	IO Parts	Result
Main Module PortA	Communications	7-Segment Display	Passed
		LED Matrix	Passed
		Analog Input	Passed
Main Module PortB	Communications	7-Segment Display	Passed
		8-Bit LED Indicator	Passed
		LED Matrix	Passed
Main Module PortC	Communications	7-Segment Display	Passed
		8-Bit LED Indicator	Passed
		LED Matrix	Passed
		Speaker	Passed
Main Module PortD	Communications	8-Bit Switches	Passed
		Keypad Matrix	Passed
		Speaker	Passed
Main Module PortE	Communications	Analog Input	Passed
		8-Bit LED Indicator	Passed
		Speaker	Passed

C. System testing

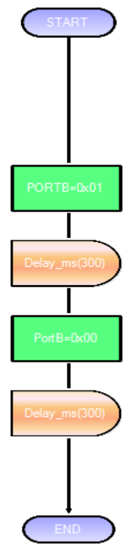
The system testing procedure follows the AGCPIC steps which include the problem definition, creation of schematic diagram, creation of flowchart and project testing. The input devices and output devices testing follows the same steps. The following steps are sampled testing that was conducted by the researcher to perform the system testing.

Step 1: Problem definition. The LED0 of the 8-Bit LED must perform the blinking function.

Step 2: Creation of schematic diagram. Fig. 15a shows the created schematic diagram of LED0 of 8-bit LED using the Auto-Generated Code for Programming PIC16F877A (AGCPIC) software. IC1 is the PIC16F877A microcontroller, CN1 is used as programming terminal to download the source code inside the microcontroller unit, Vcc is connected to 5V power supply and MCLR switch is the Master Clear that is used to reset the microcontroller. LED0 is connected to RB0 of the microcontroller. The interfacing was performed by the user using the software.



(a)



(b)

Figure 15.LED0 - Blinking Function (a) Schematic Diagram (b) Flowchart

Step 3: Creation of flowchart. Fig. 15b shows the created flowchart using the AGCPIC software. The flowchart has Start, which indicates the start of the program , the process blocks, which indicates that process that need to be performed and the End, which indicates the end of the program.

Step 4: project testing. In order to test the project, the source code that was generated by the AGCPIC software will be transferred to MikroBASIC compiler. Fig. 16 shows the screen shot sample of the generated source code of the LED0 with blinking function made using the AGCPIC software.

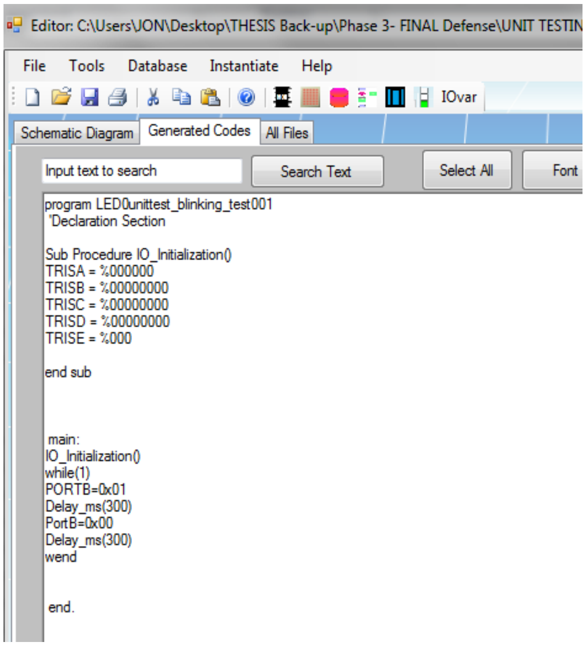


Figure 16. Sample Generated Source Code of the LED0- Blinking Function

The mikroBASIC compiler was needed only in order to produce to HEX file or the machine language of the program. Fig. 17 shows the created HEX file using the MikroBASIC compiler. The HEX file is the machine language equivalent of the source code created by the user.

The HEX file that was created using the MikroBASIC compiler is downloaded to PIC16F877A microcontroller using the Microchip PICkit 2 hardware programmer. Fig. 18 shows that the machine language was downloaded successfully using Microchip PICkit2 hardware downloader.

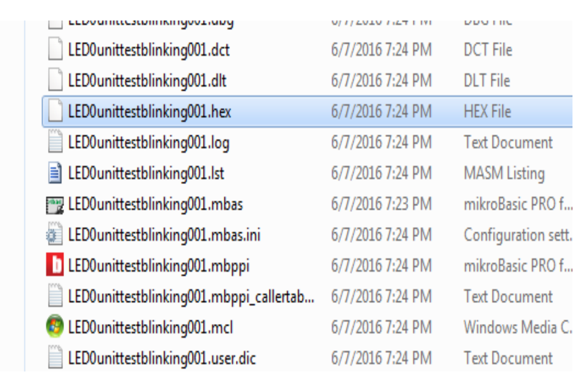


Figure 17. Created Hex File Using MikroBASIC Compiler

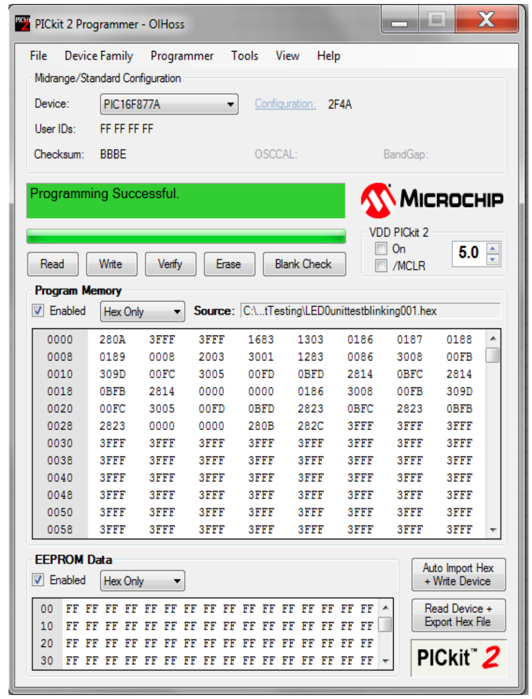


Figure 18. Downloaded Machine Language Using MicroChip PICkit2

X. CONCLUSIONS

Based on the finding of the study, the following conclusions were drawn:

The study simplifies the microcontroller unit programming, particularly on PIC16F877A programming using the AGCPIC system with the combination of textual and visual programming technique.

The study also simplifies the procedure on developing a microcontroller unit based project from eight steps to four steps. The first step is problem definition, second is creation of schematic diagram, third is creation of program flowchart and fourth is project testing. The software also provides printable schematic diagram, flowchart and equivalent generated codes needed for building an embedded system project or the microcontroller based projects.

The study also has an advantage on testing the embedded system source codes. The generated codes using the AGCPIC software were tested on the actual hardware which is called the AGCPIC Development Board. The development board is composed of main module and input-output module that the user can use to test for the validity of the codes and to check for actual response of the project.

The auto-generated code for programming PIC16F877A microcontroller reliable based on the testing performed on this study. The AGCPIC system is also “highly acceptable” system for Microprocessor System subject and for building an embedded system project.

XI. RECOMMENDATIONS

Based on the foregoing finding and conclusion, the following recommendations are presented:

1. The Auto-Generated Code for Programming PIC16F877A Microcontroller is recommended to be used by courses with Microprocessor System subject and other colleges with equivalent subject.
2. The AGCPIC system utilized interactive application for schematic design and flowcharting. The system also has actual hardware that is ideal for testing and experimentation. With these, the AGCPIC recommended to be used in developing a microcontroller based project that requires schematic diagram, flowchart and source code.
3. This study recommends for future researcher to develop a built-in compiler that converts the generated source code into machine language or the HEX file that is needed by the actual hardware.

REFERENCES

A. Books

Alciatore, D. (2011). *Introduction to mechatronics and measurement systems*. McGraw-Hill Education.

Baroth, E., & Hartsough, C. (1995). *Visual Programming in the real world*. Prentice-Hall.

Bates, M. P. (2008). *Programming 8-Bit PIC microcontrollerin C with interactive hardware simulation*. Newnes.

Gardner, N. (2002). *An introduction to programming the microchip PIC in CCS C*. Bluebird Electronics.

Maini, A. K. (2007). *Digital electronics: Principles, devices and applications*. John Wiley & Sons Ltd.

Matic, N. (2001). *BASIC for PIC microcontrollers*. MikroElektronika.

McGraw-Hill Companies, Inc. (2003). *Scientific and technical terms*(6th ed.). McGraw-Hill.

Microchip Technology Inc. (2003). *PIC16F87XA Datasheet*. Microchip Technology Incorporated.

Predko, M. (2007). *Programming and Customizing the PIC Microcontroller* (3rd ed.). McGraw-Hill/TAB Electronics.

B. Journals

Whitley, K. N. (1996). Visual programming language and empirical evidence for and against. *Journal of Visual Language and Computing, 8*(1).

Wegner, P., & Silverberg, J. (1999). Interactive visual programming: Principles and examples.

Das, B. (2010). Effect of graphical method for solving mathematical programming problem. *Daffodil International University Journal Of Science And Technology, 5*(1).

C. Publications

Jones, R. L., Hsiung, S. C., & Raeisi, R. (2012). *Introduction to a Microcontroller Training Platform for Distance Education*. Proceedings of the 2012 ASEE PSW Conference.

D. Internet Sources

Boshernitsan, M., & Downes, M. (2004). Visual programming languages: A survey. Retrieved from http://digitalassets.lib.berkeley.edu/techreports/ucb/text/CSD-04-1368.pdf

Janssen, D., & Janssen, C. (2014). Techopedia. Retrieved from http://www.techopedia.com/definition/3411/platform

Koltow, D. (2013). Description of Visual Basic. Retrieved from eHow Tech: http://www.ehow.com/about_6677935_description-visual-basic.html

Loredana, I. (2010). Visual Programming and Modeling. Retrieved from ebookbrowse: http://ebookbrowsee.net/visual-programming-and-modeling-pdf-d266337894

Ryan, V. (2010). PIC Microcontrollers. Retrieved from Technologystudent.com: http://www.technologystudent.com/pics/picgen1.html

Swanson, W. (2001). Assembly Language. Retrieved from Swanson Technologies: http://www.swansontec.com/sprogram.html