Assembly program analysis essay

ASSIGN BUSTER

My involvement in the group project was to write the assembly program. I wrote Instructions to enable the computer to communicate with the controller and do specific tasks, e. g. move forward, left or right. In the very early days of computing, a computer could only execute instructions such as machine code, which were instructions coded in 1s and 0s, which is a hard and long process. Assembly language was developed to eliminate the need to use binary codes for each instruction. Assembly language use mnemonic codes such as ADD, SUB and BRQ etc.

I decided to use Assembly language rather than C, because I needed to write a program that will execute very quickly, and use very little space as possible, because the ROM/RAM size on the 8051 is very limited. I could have used C, which would also allow me to communicate with the serial port and perform various actions, but wasn’t keen on the actual program, and structuring. I found assembly language much easier to work with and understand because I could manipulate bits and bytes of the microcontroller.

Assembly language is a low level language and is very difficult to learn and understand, therefore I had to carefully plan my code and go through various stages of planning to get to my final piece of code. I will show you how I got to the final piece of code through various stages. Diagram of 8051 Microcontroller This is a diagram of the 8051 microcontroller that we are using for the robot. You can see the various connections that are going to be made to the controller.

The two stepper motors are connected to various pins from P1. 0 to P1. 7, each pin is connected to a different coil on the stepper motor. Each pin is

connected to different coils so we can energise each one at different times, which will move the motor. I will explain this in greater detail later. We are receiving commands through the serial port, and not transmitting, therefore we are concerned with Pin 3. 0, which is RXD, which is a receive pin, which gives full duplex operation. Flowcharts This is a flowchart to show how different actions are to be performed.

This flowchart allows us to see what steps need to be taken to design the program and gives us a greater understanding. Pseudocode The next step was to design the code using Pseudocode. Pseudocode provides a means of expressing algorithms without worrying about syntax of a particular language. I have broken it down in three steps, with stage 1 being the most ‘ easiest’ form, and stage 3 being the most comparative to the final piece of code. The program is built up during the different stages and provides us with template to use when writing the code.