**AI matrix**

**Datasheet**

Written by

Dexter R Shepherd

Contents……………………………………………………………….

Overview……………………………………………………………………………………………

Summery……………………………………………………………………………………………

Schematics and reference design……………………………………………………….

Power………………………………………………………………………………………………..

Memory…………………………………………………………………………………………….

Input and output……………………………………………………………………………….

Communication…………………………………………………………………………………

Physical characteristics………………………………………………………………………

Creators of the project………………………………………………………………………

Legal protection………………………………………………………………………………..

Overview

The AI matrix is a system which collects an input, checks its data and outputs the found string of words to the user. If the input is not found in the data, the AI will ask the user to add it.

The input will then be in the solid state memory and accessible in the future when the AI is faced with this input. The output can be from saying words to physically learning through potentiometers and servos.

The AI has artificial moods, for example; if you use negative vocabulary the AI’s mood variable will be equal to sad, this means it will not have access to the main memory but have access to the sad part of its ‘brain’.

There are two types of this system:

Type one is a system with built in voice command capabilities, the ability to edit data in program such as recognized words from the voice recognition shield, navigate the algorithm on a screen using a button and diode. This feature gives you options to add words, WIFI, phrases to be recognized ect… the system can learn action or speech capabilities. What does this mean though? Well this means that you can decide whether the AI is going to say something in return to your input, or physically do something with servos.

Type two is a basic AI matrix using an earlier version of the AI software but edited to bring it into the current SHEPYANG era. This version allows, like the previous, follows the fundamental AI algorithm developed by SHEPYANG. The difference being is that this model does not use the voice control or screen. It is a basic “input -> think -> output” series.

What are the uses of this technology?

The type one is for specific robotics. Having an interactive AI you can teach things to, and talk to it as you would a human.

Type two has very educational uses. People learning code could create self-learning projects. People could use this in their own projects. Teachers could make games by adding questions to it using the SHEPYANG keyboard. Students could make little cities and use sensors to control what happens.

We hope that our AI is the future of technology and it can give the opportunity for people to teach and adapt this system for their own projects.

Summery

|  |  |
| --- | --- |
| Microcontroller | ATmega2560 |
| Operating Voltage | 5V |
| Input Voltage (recommended) | 7-12V |
| Input Voltage (limits) | 6-20V |
| Digital I/O Pins | 54 (of which 14 provide PWM output) |
| Analog Input Pins | 16 |
| DC Current per I/O | Pin 40 mA |
| DC Current for 3.3V | Pin 50 mA |
| Flash Memory | 256 KB of which 8 KB used by bootloader |
| SRAM | 8 KB |
| EEPROM | 4 KB |
| Clock Speed | 16 MHz |
| SD card memory (given) | 8GB |
| SD card memory (maximum) | The is no maximum |
| External power supply (min) | 6V |
| External power supply (max) | 12V |
| Minimum operating temperature |  |
| Maximum operation temperature |  |

Schematics and reference design

Power

Type one uses a main power supply of around 5.1V – 6V (USB connection) and a secondary power supply of 6V minimum (4 AA batteries).

Type two uses 6V – 12V through any means of input (USB or Power supply).

Memory

The main memory of both types uses an SD card. The SD card can be from any memory but it is recommended to replace the given SD with an SD with a larger size when doing big projects.

The SD **MUST** contain the readable files for the AI. The AI cannot create memory, only add to it. These files are: negative.txt, positive.txt, memory.txt, lines.txt, lines1.txt, lines2.txt. if you are using type one you will need to include all the files for the VC words and the Bitmaps for the user interface.

Input and output

Type one has a set input and output. This consists of voice input and audio output. Because the AI only understands words the audio input is converted down then uses text to speech modules for output. The AI has an extended output to servos and an extended input of potentiometers. This is extended because it’s on a separate board of the algorithm.

Type two has no built in input, instead it waits for i2c communication to pick up a string and register a character variable of ‘/’. This then tells the AI it is ready to process the sentence. The i2c communication channel is on com 2 and the AI board on com 1.

The input code will use the command:

*Wire.beginTransmission(1); // transmit the main AI*

*Wire.write("this is the string you want to send ");*

*Wire.write(‘/’); // sends the end command*

*Wire.endTransmission(Boolean true); // stop transmitting*

It is also acceptable to send string variables using loops.

*While(counter <= string.length)*

*{*

*Wire.beginTransmission(1); // transmit the main AI*

*Wire.write(string[counter]);*

*Wire.endTransmission(Boolean true); // stop transmitting*

*Counter += 1;*

*}*

*Wire.beginTransmission(1); // transmit the main AI*

*Wire.write(‘/’); // sends the end command*

*Wire.endTransmission(Boolean true); // stop transmitting*

The output works on com 3 of the i2c communication. It will send a string and a command character of ‘#’. It is for the programmer to decide what to do from here. The AI will send out the variable of ‘@’ if a function needs to be added. This is when the output board needs to tell the user that there is no data for the input. This alerts the user and when the input the system will save.

Example code:

*void receiveEvent(int howMany) {*

*char x = Wire.read(); // receive byte as an character*

*Serial.println(x); // print the byte*

*If(x == ‘#’)//output*

*{*

*Serial.print(string);*

*string = “”;*

*}else if(x == ‘@’) //learn*

*{  
Serial.print(“learn mode”); // this can be output to the user’s preference using text 2 speech modules ect…*

*}else{*

*string += x;*

*}*

*}*

Your only limit is your imagination with the inputs and outputs. You are able to create input/output through any means. You might even create your own voice control. The input doesn’t just have to be on one board, you might have hundreds of boards gathering data and sensing them.

The board does have a built in output, you can control pin output. This means when talking to the robot you can teach it to send 5v of power out many of the GPIO pins. This could be used in sensors where it sends commands to the main AI board where it will control lights or motors. This is an inbuilt extra function the AI is capable of learning. The AI can control any of the characteristics of its physical state. It just needs to learn how and when to use them.

Communication

As previously mentioned, the AI is on the i2c com 1, the input board on com 2 although it can be changed, the output board must be on com 3.

Physical characteristics

The board contains an Arduino Mega clipped on to the back to do all the processing. The AI shield has an SD card wired to the Arduino, LEDs to show different things and lots of GPIO pins, some for board communication, and others for different AI types.

Type one will have a TFT screen soldered on, it will not have an SD module on as the screen has a built in one. Type one also has GPIO pins which are for user integration. This meaning the user can add certain text to speech modules if they wanted a different voice.

Type two is all about adding what you want instead of going with recommendations. The board has prototyping capabilities to make your own circuit.

Creators of the project

The project started in 2015 although it was not the AI yet. In 2015 a year 10 group at PACA school started creating games which could learn questions rather than have them put in. these games were interactive using RFID scanners to answer the questions. This project lead on to the development of Artificial Intelligence in 2016.

Dexter Shepherd, the joint group leader and primary software developer, brought his theory and idea of the AI to the group on a Facebook group chat. The group liked it and was interested in what he’d created using an Arduino Uno. This sparked the AI project and led to the creation of SHEPYANG.

Guolin Yang, the joint group leader and web developer, has worked on the AI contributing ideas and code. Guolin made a c++ version of Dexter’s Arduino AI. Guolin developed the website and manages relations with other groups and companies.

Josh Hassett, the main hardware development, puts together the groups designs and finds effective ways of development and assembly.

Morgan Pile is the groups designer. Morgan developed the logo for SHEPYANG and designs all the casing/hardware for the AI using CAD for 3D printing.

Legal protection

This is yet to be seen.