

**PROJECT TITLE:** *Pin 3D Display Panel*

**TEAM MEMBERS:**

*Gregor Doler, [gregor.doler@uni-mb.si](mailto:gregor.doler@uni-mb.si)*

*Matjaz Boldizar, [matjaz.boldizar@uni-mb.si](mailto:matjaz.boldizar@uni-mb.si)*

*Miha Grm, [miha.grm@uni-mb.si](mailto:miha.grm@uni-mb.si)*

*Jozef Plavec, [jozef.plavec@uni-mb.si](mailto:jozef.plavec@uni-mb.si)*

**ADVISING PROFESSOR:**

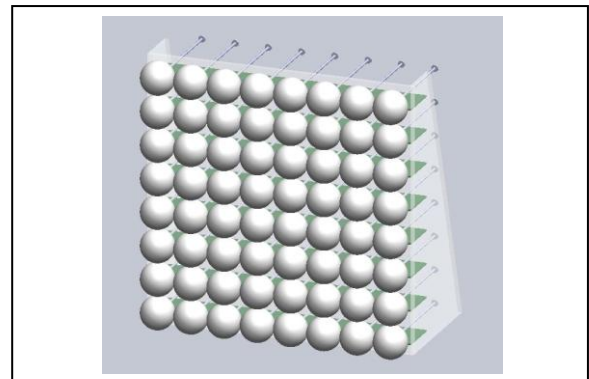
*Iztok Kramberger, [iztok.kramberger@uni-mb.si](mailto:iztok.kramberger@uni-mb.si)*

**UNIVERSITY:** *University of Maribor, Faculty of electrical engineering and computer science, Smetanova 17, 2000 Maribor, Slovenia*

**DATE:** *25/06/2010*

**TI PARTS USED IN PROJECT:**

- 1, **TUSB3410**, <http://focus.ti.com/docs/prod/folders/print/tusb3410.html>
- 1, **MSP430F2272**, <http://focus.ti.com/docs/prod/folders/print/msp430f2272.html>
- 8, **MSP430F2132**, <http://focus.ti.com/docs/prod/folders/print/msp430f2132.html>
- 64, **DRV8811**, <http://focus.ti.com/docs/prod/folders/print/drv8811.html>
- 1, **TPS40195**, <http://focus.ti.com/docs/prod/folders/print/tps40195.html>
- 1, **TPS54325**, <http://focus.ti.com/docs/prod/folders/print/tps54325.html>



## PROJECT ABSTRACT

The purpose of our Pin 3D display panel is to be able to produce three dimensional figures by moving single pixels forward and backward depending on it's position in a 3D model which is generated on a personal computer (PC). Our main goal was to be able to create an array of 32x32 pixels. For the "pixels" we would use table-tennis balls which have 40mm in diameter. The end result would be a wall of 32x32 balls (1024 all together) in a 160x160cm (approx. 63x63 inches) area.

For the competition purpose we decided to build only 8x8 array because it can work as a standalone unit, therefore additional modules are more or less *plug-and-play*, are built the same way and can be added easily.

We have decided that the best way to move the balls back and forth would be by using a stepper motor. So we have designed our own unique version of the motor that we named "linear stepper motor". But due to a large number of motors needed to build such display panel we were unable to find a company that would build them for free so we had to compromise and use one of manufacturers already designed motors which has many more steps and is therefore more precise depth-wise. These new motors are sadly a bit slower than our expectations were.

Our project covers all the hardware and software that is needed to get a 3D figure from a single USB port on a personal computer.

- **Motivation For Project**

We got our idea for this project from so called “Pin Art” which is basically an array of pins that can be moved from the back side by pushing in an item of some sort. This item is then displayed in the front until we move the pins back in.

This concept is originally totally mechanical and we could not find any product on the market that is controlled electronically. So we found that it could be an interesting project and something that has never been done before.

- **Theoretical Background**

Pin 3D display panel will consist of 64 pins (table-tennis balls) arranged in 8x8 array. Each single pin is controlled by an individual stepper motor, which moves the pin forwards and backwards. This is how we get desired 3D effect.

The motors are driven by stepper motor drivers which have integrated H-bridges. These drivers are controlled by a slave microcontroller. Every microcontroller controls 8 motor drivers (one row of motors).

Slave MCU has the task to control, maintain, and show data for it's row of pins (eight motors). In certain intervals, slave MCU must reset the position of the eight controlled pins – it moves them to their reference point. We have chosen the reference point to be when the pin is in its most backwards position.

Master MCU accepts data from PC's USB port and forwards data/commands to slave MCUs, via I<sup>2</sup>C interface. Data from PC is transmitted by UART.

- **Implementation**

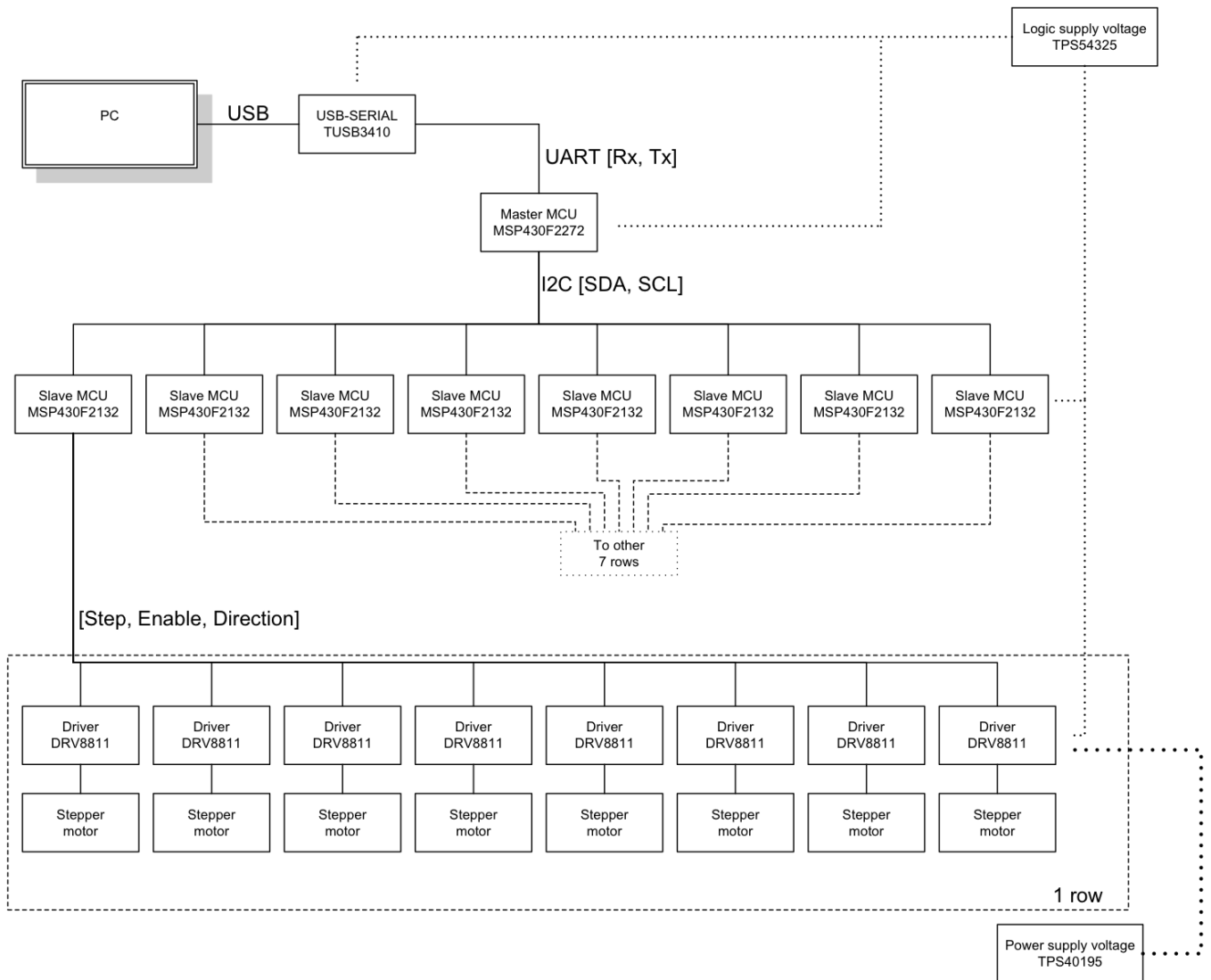
For the basis of our project we had to figure out, how to design and control one module - 8x8 pins. PC will generate data from digital media (animations, texts, simple movies, etc.) and transmit it to master microcontroller. Because there is no MSP430 with USB support available at this time, we have chosen USART protocol. Because of that we also need TUSB3410 module for USB to serial conversion.

- **TUSB3410** provides bridging between a USB port and an enhanced UART serial port. The TUSB3410 contains all the necessary logic to communicate with the host computer using the USB bus. It contains an 8052 microcontroller unit with 16K bytes of RAM that can be loaded from the host or from the external on-board memory via an I<sup>2</sup>C bus. It also contains 10K bytes of ROM that allow the MCU to configure the USB port at boot time. The ROM code also contains an I<sup>2</sup>C boot loader. All device functions, such as the USB command decoding, UART setup,

and error reporting, are managed by the internal MCU firmware under the auspices of the PC host.

- For master MCU we have chosen **MSP430F2272**. It contains 16-bit RISC CPU, 16-bit registers, and constant generators that contribute to maximum code efficiency. It has sufficient amount of RAM (1 KB), 32 KB of flash and all peripherals that we need (USCI\_A(UART), USCI\_B (I2C), 2 16-bit timers). Its main task is to communicate with a PC, and send data to slave devices. One 8x8 matrix has one master device and 8 (1 per row) slave devices.
- Slave devices are **MSP430F2132**. They also contain 16-bit CPU, 16-bit registers, and constant generators that contribute to maximum code efficiency. They have 512 B of RAM, 8 KB of flash memory, 24 GPIO, USCI\_B (for I2C), and 16-bit timer. Their main responsibility is controlling DRV8811 accordingly to data they obtain from master MCU.
- For stepper motor driver we have chosen **DRV8811**. The DRV8811 provides an integrated stepper motor driver. It consists of two H-bridge drivers, as well as micro-stepping indexer logic to control a stepper motor. The output driver block for each consists of N-channel power MOSFETs configured as full H-bridges to drive the motor windings. A simple step/direction interface allows easy interfacing to controller circuits. For our circuitry we will be have common step, and separated enable and direction for each DRV8811. With that we will achieve easier and smoother movement of pins. Pins allow configuration of the motor in full-step, half-step, quarter-step, or eighth-step modes. Decay mode and PWM off time are programmable. We will be using it in full-step mode. Internal shutdown functions are provided for over current protection, short circuit protection, under-voltage lockout and over-temperature.
- To get a stable and efficient power source for the digital power supply we have decided on using **TPS54325EVM**. It is a wide input voltage step-down switcher with integrated FET. It is capable of delivering up to 3A output current which is enough to power all of our modules simultaneously. For our needs we have set the output voltage to 3.3V which can be applied to MSP430 MCUs, TUSB3410 USB-to-serial converter and DRV8811 (logic supply voltage).
- Because of higher energy consumption and higher input voltage of the stepper motors we had to use additional power supply. We have tested DRV8811 on laboratory power supplies and came to a conclusion that we could use **TPS40195**, synchronous buck controller which can deliver up to 20A output current. For our testing stepper motors we needed to set 12V output voltage which is well within TPS's voltage output range. One TPS40195 will be sufficient to power up one module at a time. Large capacitors must be used with each DRV8811 for compensation of high current surges while operating stepper motors.

All together will be connected as shown on lower diagram:

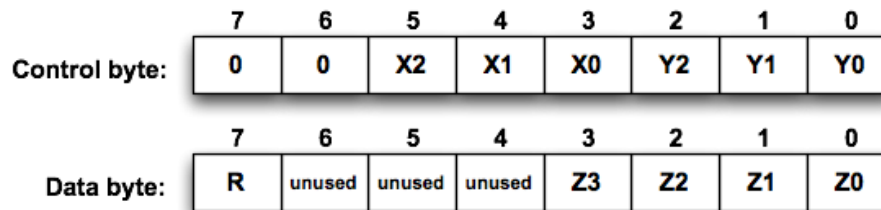


**Figure 1: A block diagram for one 8x8 module**

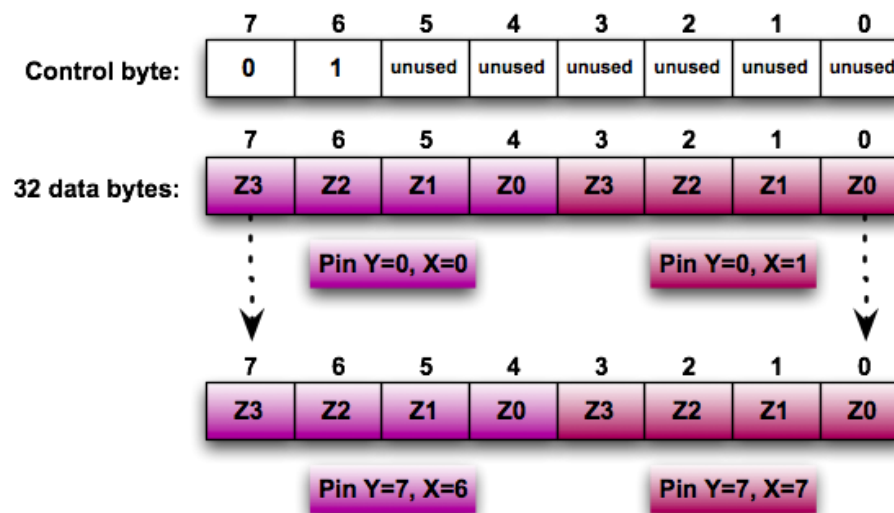
Description of communication protocols.

## UART communication definition:

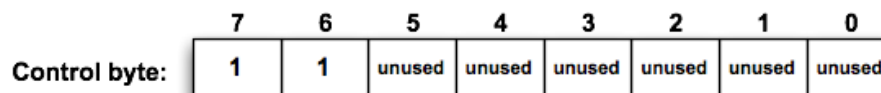
### 1. PC sending a position of specified pin



### 2. PC sending a position of all pins



### 3. PC sending reset request for all pins



**Figure 2: UART communication protocol diagram**

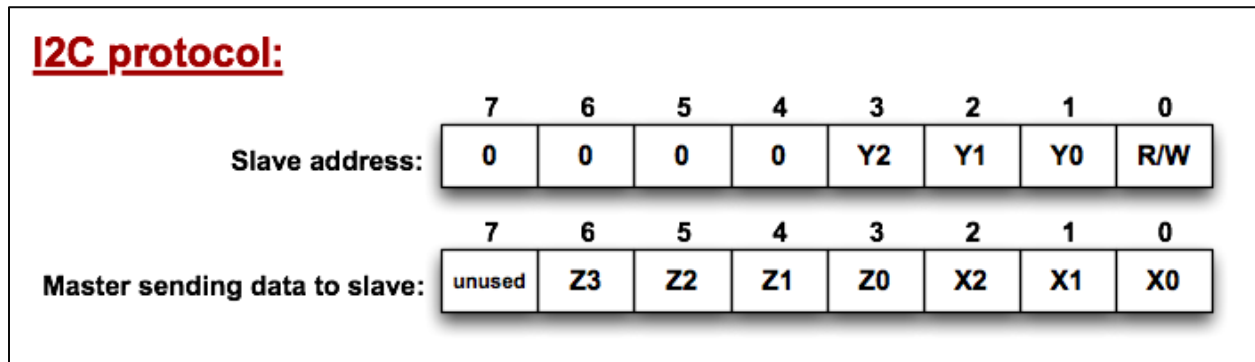
PC and master chip will communicate via UART. Baud rate of communication is 125000 Bd, with no parity checking, and no handshaking.

There are three possible data packages:

- PC sending a position of a specific pin
  - In first, control, byte PC sends X and Y position of pin in 8x8 matrix
  - In second, data, byte PC sends Z position of a pin, or a reset command for pin.
- PC sending a position of all pins

- In first, control, byte PC sends command for sequentially sending Z coordinates
- In next 32 bytes, PC sends coordinates for two pins at once - in first byte we have Z coordinate for first and second pin, in second byte we have Z coordinate for third and fourth pin, and so on.
- PC sending a reset request for all pins
  - PC sends only control byte, with command for reset all pins.

Master and slave MCU communicate via I<sup>2</sup>C bus. We have selected I<sup>2</sup>C because of simple usage with MCUs. We will be using high speed mode (3.4Mbit/s).



**Figure 3: I<sup>2</sup>C communication diagram**

In 8x8 array, we have eight slave MCUs. Each one has I<sup>2</sup>C address respectably to its row number.

Communication on I<sup>2</sup>C is very simple - first master sends address (row number), and then data - Z and X coordinate.

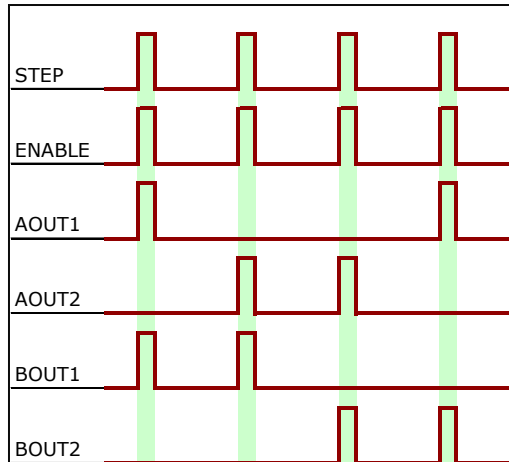
For transmitting one frame for whole 8x8 matrix PC must send 33 bytes to master MCU. For sending 24 frames per second (for displaying video) minimal bandwidth must be 792 bytes per second. With our speed (125000 Bd) we can easily achieve that.

For transmitting one frame from master to slave MCU, master must send 128 bytes (64 bytes of addresses and 64 bytes of data) to all slaves. That's 3 Kbytes at 24 frames per second. That is also achievable with our I<sup>2</sup>C speeds.

For showing one frame, total of 161 bytes must be sent and for showing video at 24 frames per second a total of 3.77 Kbytes per second must be sent.

### • **Experimental Results**

Each stepper motor draws 50mA peak current for each step. This means that one 8x8 module draws a maximum of 3.2A current if all 64 motors were stepping simultaneously. These currents are existent only at step impulses. Because our sample motor does not need to be held in place, no current is needed while the motors are not in movement.



**Figure 3: Example of step impulses**

With the sample motor that we got for testing we have measured that the maximum speed is around 1000 steps per second. Each plant consists of 48 steps which gives us 4.8 seconds for the motor to travel completely from the back to the front (160mm). The calculations are as in the figure below:

stepper motor		1000 max_step/s		
		48 step/plant		
distance [mm]	pitch bolt	plant/s	distance[mm]/s	time [s]
160	1,6	20,8	33,3	4,8

**Figure 4: Calculations of speed**

- **Conclusions**

Due to unavailability of DRV8811EVM evaluation module at Texas Instruments free tools we were very much setback time-wise so we had to order a sample of DRV8811 and build our own evaluation board for stepper motor testing.

We also did not get the customized stepper motors until this date so we were able to get the UART and I<sup>2</sup>C working exactly as we needed but only one stepper motor is currently controlled because of only one evaluation board and only one sample stepper motor.

If we were to use the sample motor that we are currently using it would be too slow for our needs. Therefore we have made some adjustments to the sample motor and we are now awaiting their production at a Slovenian company Iskra Mehanizmi.

- **Summary**

In the process of working on this project we have gained a lot of useful knowledge. We have found out that Texas Instruments has a MCU for every occasion. We have chosen MPS430 series for our project.

We have learnt how to program the MPS430 series and how to use integrated peripherals. We have successfully utilized MSP430's timers, basic clock module, low power modes, analog-to-digital converter, UART and I2C communication. We have also successfully learnt how to use debugging tools.

We have learnt how the stepper motor works, we built our own linear stepper motor which worked well with own DRV8811 experimental board but did not get into production because of high cost for our budget. We have then compromised for an already made stepper motor that would be cheap to modify for our needs. DRV8811 was able to drive this new motor as well with only a few minor modifications.

We have also tested the TPS54325EVM evaluation board quite extensively. We have learnt how this particular step-down switcher works. We have measured it's efficiency and stability to find that it is more than sufficient to power the complete project (16 modules) if we will be able to build it in the near future.

- **Future Plans**

As we are currently still waiting to get the stepper motors finished the future is definitely clear – we have to build the whole array of pins to get the desired 3D effect.

Apart from the mechanical part of the project, we will choose another Texas Instruments' MCU which will have two I2C interfaces with enough bandwidth to get all the information from USB to each individual stepper motor. We are also planning to use new MSP430 with integrated USB controller. This way we will be able to leave out USB-hubs and simply use only one data converter to control all the modules at the same time.

If we will be able to find funding for further development we will also try to get the motors and the pins minimized as much as possible so that an array of smaller pins could be used to display a 3D face in 1:1 ratio. We find that very interesting for video calls. Not only that one speaks to a person on the other end of the line, one can watch the other person's facial expressions and get much more information about his or hers emotional reactions.

We are also planning to add RGB LEDs to illuminate each pin on display and get colour image.