Embedded System Design Documentation

1. Target Embedded System

**Power**

The target microcontroller ATmega328p requires power input between 3V – 6.6V. We used MCP18XX regulator which gives fixed output voltage of 3.3V. This choice was done to be compatible with MSP432 SPI Port, as MSP432 operates on 3.3V.

**Why ATmega328p?**

ATmega328p is a microcontroller built on AVR RISC architecture. This platform is new to us, and this class of microcontroller is widely used in Arduino boards, for which heavy documentation is available in case there is any problem.

**What did we learn about AVR?**

ATmega328p has an AVR Core with registers. It has 32kB of Flash memory and 6kB of SRAM to use during program run. It also hosts 2kB of EEPROM for code or data storage. Flash Memory can be coded through Serial and Parallel Programming. However, we decided to go with Serial Programming as it requires less hardware support.

**ISP Protocol**

ISP Protocol for ATmega328p is simple. It is built on SPI. SPI module in ATmega328p activates as reset is active. Once reset is active, wait for 20 ms, and then send programming enable command. If the device is in sync, then it will echo back “53” which is second byte of Programming Enable Command. Subsequent commands follow suit, and the reset is held active till the programming option is finished.

1. Programmer

**MSP432P401R**

We are using MSP432P401R as a programmer. Leveraging the SPI peripheral, and understanding how it operates, we are controlling the ATmega328p pins for Reset, SCK, MOSI, MISO to emulate the programming sequence and algorithm required for ATmega328p.]

**SPI Peripheral**

SPI Peripheral in MSP432P401R is convenient, and enough documentation is provided to get it started. We went through the peripheral registers, and designed a library to initialize SPI Peripheral, send and receive data through polling.

**ISP Command Interface**

SPI library is used by ISP Command Interface which manipulates Reset, SCK, MOSI, and reads through MISO. ISP Command Interface is used by Programmer State Machine to serially download code into ATmega328p. This element has been designed to perform ISP operations on ATmega328p.

**Hex Parsing**

Hex file is transmitted by Transmitter Device through Bluetooth. This hex file shall be parsed to deduce required ISP Commands that we will be needing. This needs understanding of Intel Hex Format – which we are going to use, and memory map of ATmega328p Microcontroller.

**Bluetooth Data Reception**

We are using HC-05 Module to communicate through Bluetooth to the Transmitter. This module is used in AT mode to set the role as a slave device, and interfaces with MSP432 using UART. This module deals with data reception, and transfers the data over to MSP432.

**CRC/Checksum**

As, the data is program data, and very crucial. We have Checksum in place to detect any error in transmission. As the data travels through multiple communication channels, we thought it would be better to ensure that the data is valid by using CRC32 peripheral in MSP432.

**State Machine**

**\_\_Diagram\_\_HERE\_\_**

1. Transmitter