EECS 373 — Wearable Game Controller — Final Report

Brandon Waggoner (bwaggone@umich.edu)
Jacob Sigler (jmsigler@umich.edu)
Brian Klein (blklein@umich.edu)
Jonathan Zarger (jzarger@umich.edu)

April 18, 2017

Link to Drive with Libero and C files

1 Description

This project would involve creating a wearable video game controller that would be able to take in human input and send it to a video game. Specifically, two boards attach to each of the user's ankles and interpret physical motion as the cardinal directions for a game controller or keyboard input. Dance Dance Revolution (DDR) will be the main game of choice for demonstration on PC, but other games and computer tasks could be incorperated with little adjustment to the overall system.

2 Customer

Game developers and players, most notably for rhythm based games like Dance Dance Revolution or Crypt of the Necrodancer. Gamers who want to try a new spin on an old classic may find it interesting and fun as well. There also exists a market in virtural reality games where additional tracking capabilities may improve user experiance.

3 Value

It would provide an interesting new way to play games on PC. Perhaps in conjunction with a VR headset, it could work in tandem to detect more fine tuned leg movements, and could possibly be used outside of video games for general computing. The system can also replace more expensive and unweildy physical game controlers, such as the DDR mat, which can cost up to $200.

4 Approach

4.1 Approach Overview

The general outline is as follows: Use an IMU with a microcontroller/FPGA to capture motion events. Convert the motion events into input controls. Transmit the input controls to a computer. Finally, use input controls as keyboard or gamepad inputs into game. If time permits, use a gyroscope input as well to define other buttons or keys.

In order to determine which direction the foot is attempting to step in, we attempted to use a dead reckoning algorithm on each IMU to determine X and Y position, forcing a reset at the origin after halt in motion to mitigate the error. To determine when weve stepped, we processed the accelerometer data to smooth

out the noise, and take notice of the large spikes in accelertions. Those points were very well defined, and relatively easy to detect.

The then determined inputs were fed into a keyboard emulator through bluetooth to a receiver in python, as many games and programs support either. The receiveing computer then runs the game/game emulator which accepts the keyboard inputs.

The project was built around a printed circuit board hardware component. It featured a SmartFusion SoC in the same line as the one used in lab, but in a form much easier to solder. It had a built-in inertial measurement unit, power electronics, battery conditioning and monitoring, USB debugging, and a Bluetooth module. The board was powered by a 1 cell lithium polymer battery, which needed to be charged externally.

Initial prototyping for this device was performed with several devices the team already had in hand. Preliminary inertial measurement data collection was collected through a team member's previous project that contained the same IMU that was be used for the project. While unrepresentative, it did provide a good baseline for the type of data to expect.

4.2 Major Functions of the Project

4.2.1 Algorithm

Digital signal processing and analysis on the accelerometer data from the IMU. The processed data was put into a dead-reckoning algorithm to determine position, and from position we attempted to determine the intended direction of the foot controller. The Z-acceleration was be used to determine the moment of impact, and when exactly to send the "button press" data. We were unable to fully implement dead reckoning within the time frame, but thought of a few alternative ideas, such as hand motion, or orientation of the board rather than raw acceleration.

4.2.2 Communication

The data was be sent over Bluetooth (exact module listed in Section 5), to a computer that interpreted the data as a keyboard input or joypad press. The receiver was written in python, using pyautogui and the python serial module.

4.2.3 Board Controller

A SmartFusion board was written to interface with the IMUs and pull the data.

4.2.4 Printed Circuit Board

The printed circuit board was the platform that allowed for mechanical and electrical connection of all hardware components. It allowed the project to have a very small form factor that could be mounted on a player's shoe or leg without being bulky or unwieldy. It was a four layer board (top and bottom signal, middle ground and power) with components on top and bottom to minimize the physical size. Mounting holes were placed on the board to allow it to attach to a player through a strap. The board was designed through Altium CircuitMaker, a free version of Altium that allows for remote simultaneous collaboration so that multiple team members can work on the project easily.

5 Essential System Components

All physical components 2x as one controller is needed for each leg.

1. Processing
   (a) (have for prototyping, order for board) Microsemi SmartFusion SoC (Lab evaluation board for prototyping, A2F060M3E-TQG144 for PCB)
   (b) (have) PC to run game
2. Inputs
   (a) (order) 9 degree of freedom inertial measurement units (MPU-9250 dev board for prototyping, IC for board)
3. Outputs
   (a) (order) Bluetooth Module (HC-05)
4. Other
   (a) (design, order) Custom Printed Circuit Board
   (b) (order) Other components for the PCB. BOM to be finalized soon.
      i. MCP1700T-3302E/TT
      ii. TPS79915DDCT
      iii. DS1818R-10+T&R
      iv. LTC4412ES6#TRMPBF
      v. MAX17043G+T
      vi. CP2104-F03-GM
   (c) (order) FlashPro4 MicroSemi/Actel Programmer
   (d) (order) 1S Lithium Polymer battery
   (e) (design, order) Power electronics (battery to microprocessor power)
5. Software
   (a) Stepmania (Open source DDR PC clone)
   (b) Mednafen (Open source NES Emulator)
   (c) Bluetooth driver
   (d) C/ARM/Verilog for board
   (e) Python for computer Bluetooth connection

5.1 Description of Difficulty Points

1. 2x IMU over SPI — .5 Difficulty
2. 2x Bluetooth transceivers on board over UART — .5 Difficulty
3. Hit/Rise Detection Algorithm + Direction of Motion — 1.5 Difficulty
   (a) Hit and rise detection working well, direction of motion not. Significant attempts made to work with available data, ran out of time. Data acquisition and debugging over UART)
4. Custom Printed Circuit Board — 2.5 Difficulty
   (a) Compact, durable. Built in SmartFusion A2F060M3E, MPU9250 IMU, and HC05 Bluetooth module. 1S LiPo voltage regulation and battery measurement. Soldered on with oven, hot air, and iron. 0603 Passives. 20MHz XTAL. Supports 25MHz SPI.

5. Battery voltage chip over I2C — .5 Difficulty
6. Receiver Program on computer, reads inputs from serial port into Python to generate game keyboard inputs. — .5 Difficulty
7. Microsecond counter on FPGA, read over APB — .25 Difficulty
8. MATLAB and R data processing code for dead reckoning — .25 Difficulty
    (a) Hit detection written, simple velocity determination tests also written. This was the point we couldn't move past.

6 Diagrams

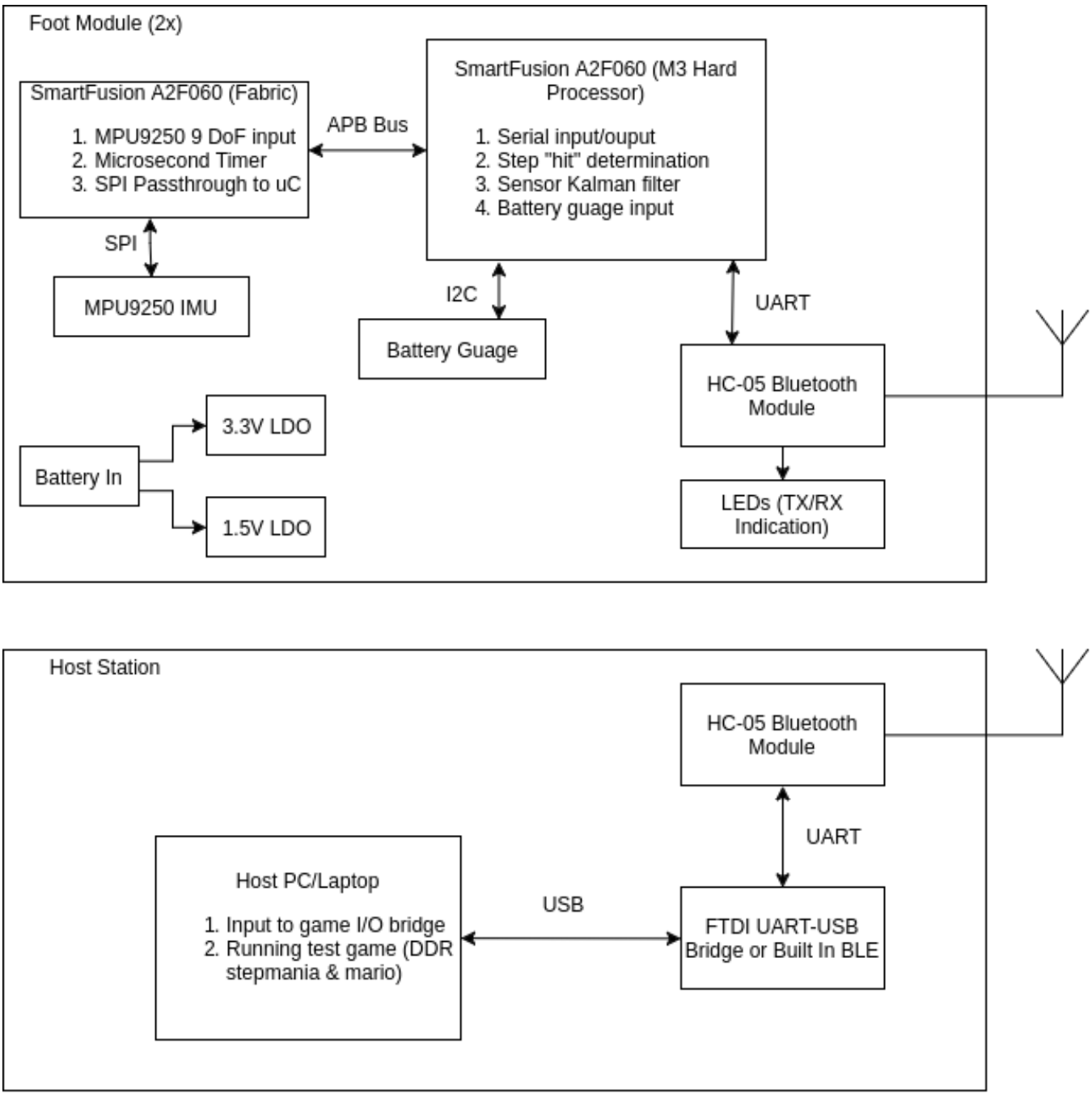

Figure 1: Component Level Diagram

The schematic capture and board design project live at this link: https://workspace.circuitmaker.com/Projects/Det
Sigler/EECS-373-DDR.

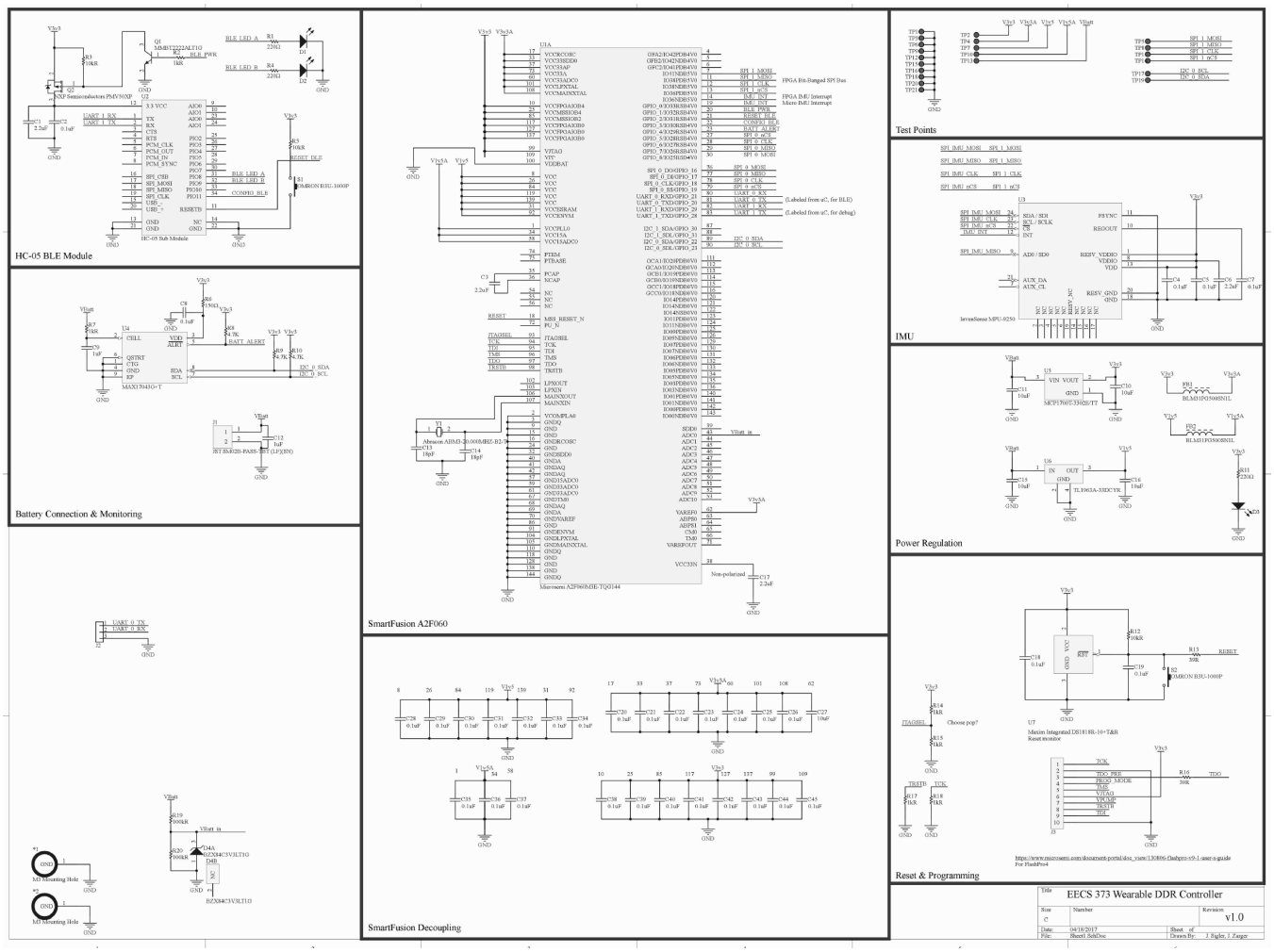

Figure 2: Schematic Capture

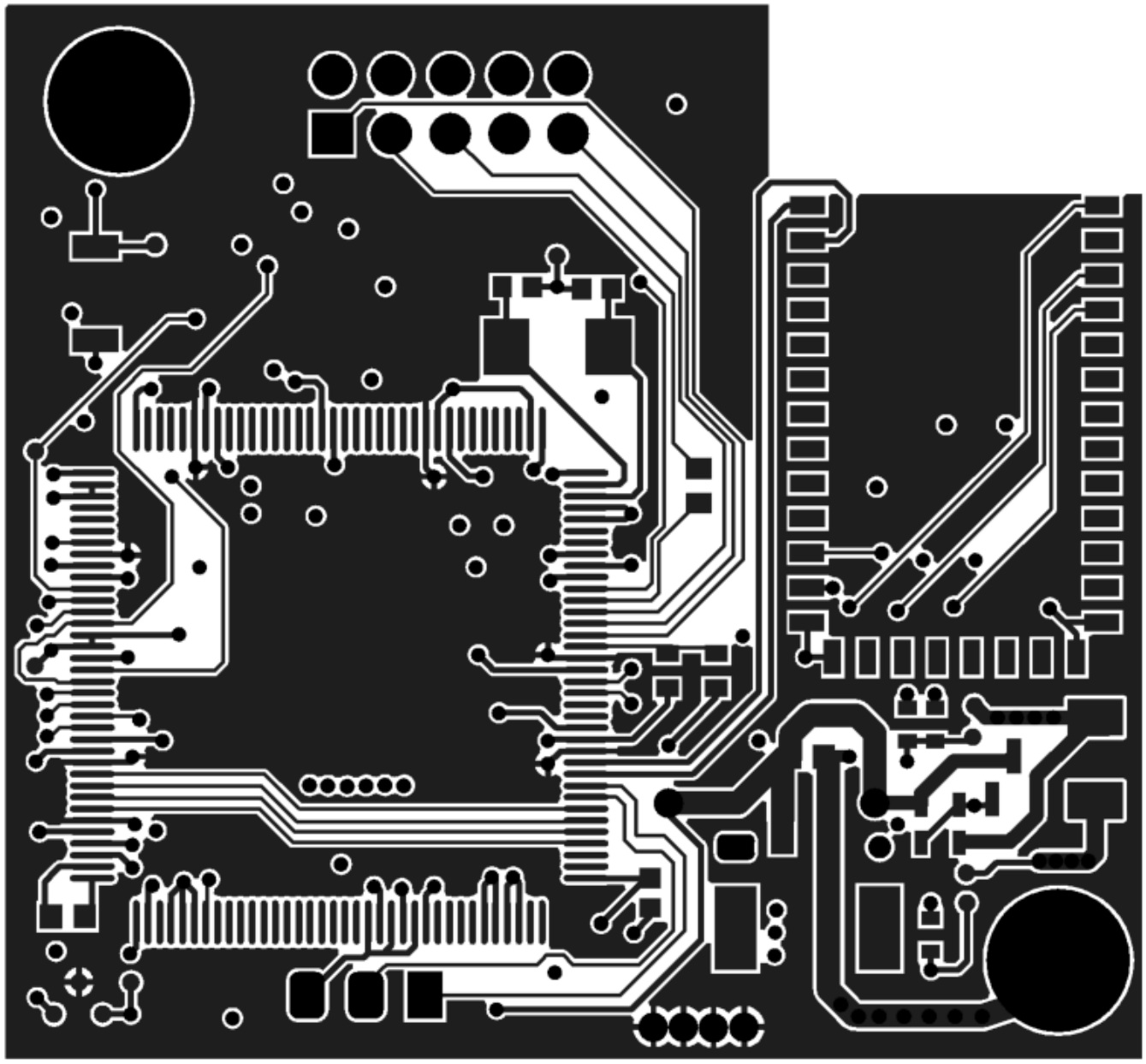

Figure 3: Board Top Layer

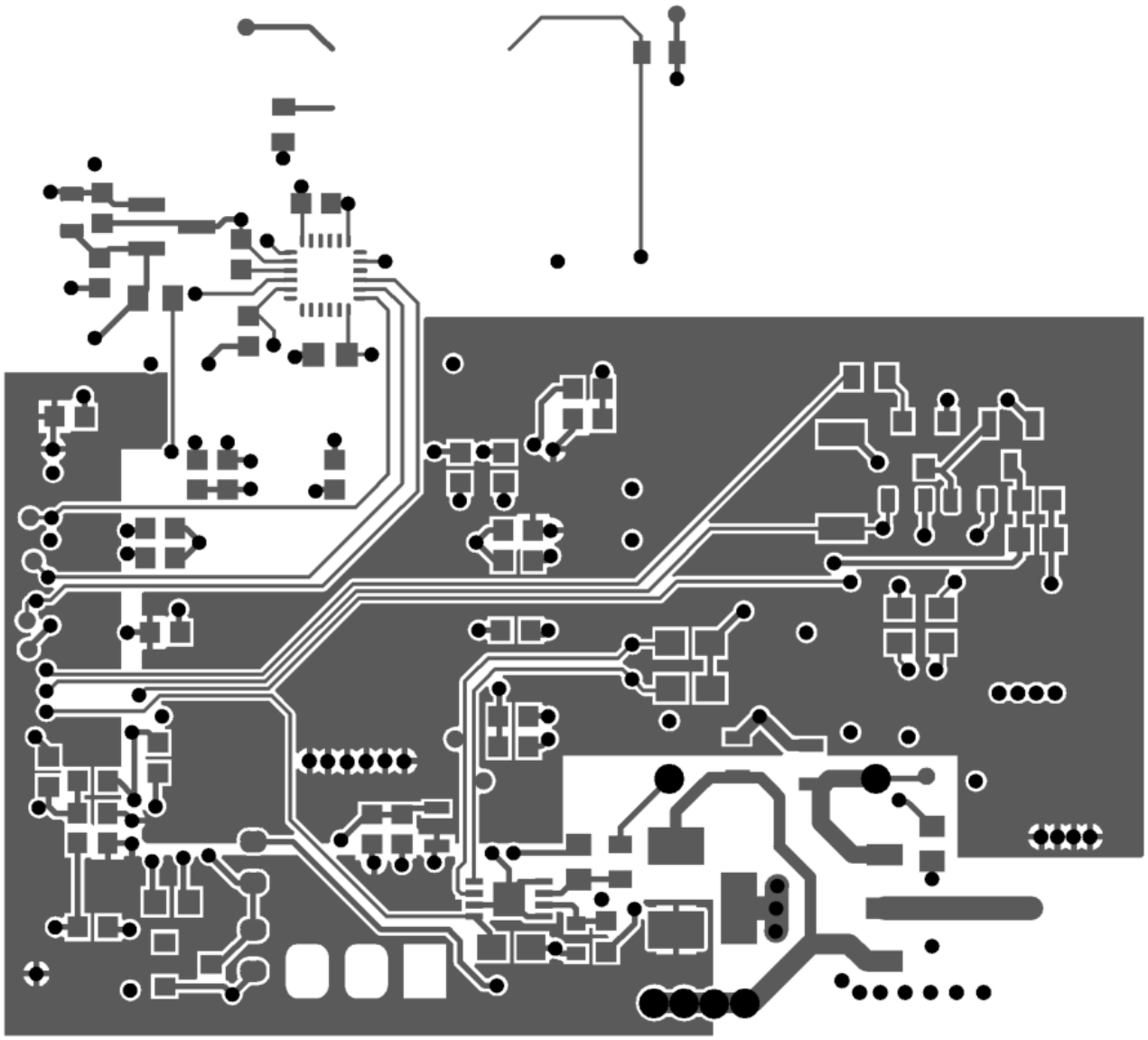

Figure 4: Board Bottom Layer

Board inner layers are just power planes.