# Read Me First

This document should act as a starting point for the continuation of P09222. The issues experienced by P09222, the changes made from the previous iteration and concepts developed that should be implemented are explained in this document. It is necessary that all team members read all sections of this paper as all issues inter-relate to other aspects of the project that will affect you.

## Engine Dynamics

Efforts were made to develop a new model to accurately predict the RPM, Track data recorded by the Motec system was used to analyze the models. This data which is collected by the Motec ECU during all engine operations is uploaded by the Formula after each driving session. This data is opened in Motec's i2 Standard Software [7] contains readings from all inputs at 1000Hz frequency. This data was manipulated in Excel to create input variables: RPM(n-1), ΔRPM(n-2), RPM(n-3), ΔRPM(n-4), as well as linear and polynomial forms of IAT, ECT, TPS, MAP and O2 to predict the output variable RPM. The variables that were insignificant were then removed from the MiniTab model to provide a lean, more accurate prediction of the RPM and therefore more accurate fuel injection and ignition. The model determined that the previous 3 RPMs and a linear coefficient of the throttle position were needed to accurately predict the next RPM.

The resulting MiniTab Model is:

Eq. 2) $\Omega_{exp=1.8+}\ \Omega_{previous} + 0.238*\Delta\ \Omega_{(n-1)} + 0.234*\Delta\ \Omega_{(n-2)} + 0.0466*TPS$

This model was compared against data for the entire fifteen minute run and results proved to be slightly better than the model currently being used in code. However, all exceeded the 0.5 degree error allowance in the specifications.

The current model in use is:

Eq. 1) $\Omega_{exp} = A\acute{\alpha} \pm \sqrt{}(A^2\acute{\alpha}^2 + 2B\acute{\alpha} + \omega_n^2)$

The optimization of this model can be tested further by determining values of constants A and B in equation 1 above. This modeling can be done by determining the linear relationships with A and B and analyzing Formula runs against those to fine optimal values for A and B to predict the RPM. In this equation, omega (Ω, ω) is RPM and alpha (ά) is acceleration determined by the previous change in RPM.

For further linear regression modeling, consult Dr. Brian Thorn in the Industrial Engineering department.

With the model optimized, additional accuracy can be obtained through mechanical changes to the car.  The greatest opportunity for improvement comes in replacing the current two-tooth cam wheel with an n-tooth wheel would allow the software to calculate RPM n times per cycle. This in-turn requires greater processor performance on the PCB to run the calculations for every tooth sensed.  Note that wheels with a greater number of teeth are very common.

## Hardware Issues

The ECU as designed can operate within the Formula Team's marginal input voltage range. The +8Vdc voltage regulation fails at the lower end of the ideal voltage range. This system does not power any circuits on the board, but is used to power an external device on the formula car. The existing +8Vdc voltage regulator may need to be replaced with a buck-boost converter to meet this specification. This is not high priority as the board itself can still function at the low end of the ideal specification and the marginal input voltage range is still satisfied by the present design.

- The input voltage fuses are still experiencing failure during operation, mainly the +12Vdc input fuse. We believe this is caused by a slight spike in current as the board is turned on. There are two possible solutions:
    - Increase the fuse size; 500mA may be too small
    - Switch from a fast-acting fuse to one with a longer time lag so it does not become damaged by the initial current spike when the ECU is powered up (recommended).

- The voltage regulator for the microprocessor (TPS70302) is the adjustable voltage regulator that was selected for use while we were still testing the board functionality. It should now be replaced with the fixed +3.3/+1.9V counterpart, the TPS70351. We already have samples of this part in stock (see Prof. Slack for the complete part supply).

- The microprocessor, a TMS470R1B512PGET, is being phased out of production by Texas Instruments. Contact TI and try to find a recommended replacement part that is able to operate within the high speed & accuracy specifications for the ECU. An ideal replacement would have a larger cache size.

- The relay control scheme should be functioning properly. If it begins to fail during operation, look into implementing a "totem-pole" control scheme. The MOSFET transistors may not be receiving a high enough gate voltage to turn them ON.

- The O2 Sensor was designed to operate with an FDS6612A MOSFET transistor, which has been phased out of production. The FDS8884 transistor used on the relay control circuits should be a simple substitution for the FDS6612A, but this will need to be confirmed.

- The O2 sensor interface circuitry has been added to the PCB and tested using PSpice software. The ECU software that uses the O2 sensor data still needs to be written. Once the code is written then the hardware can be verified. Figure 1 shown below is the O2 sensor interface circuitry that is currently on the PCB.

## O2 Sensor Circuitry

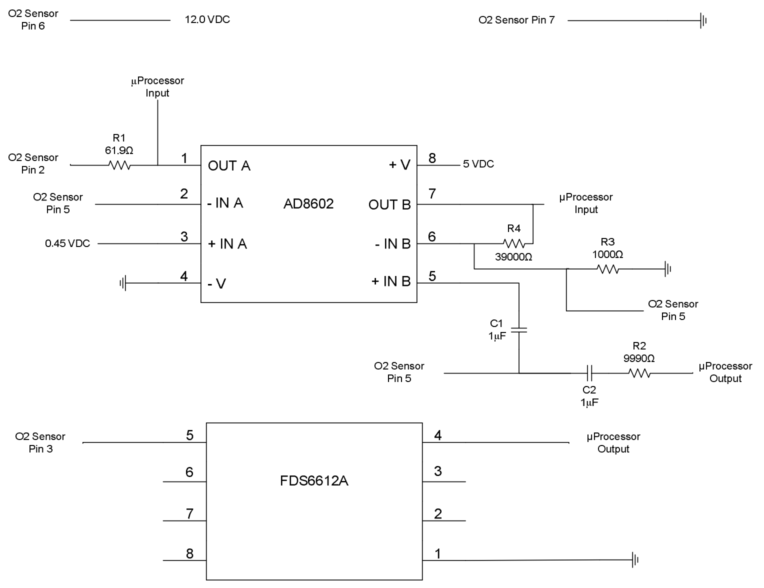

**Figure 1 O2 Sensor Interface Circuitry**

The addition of the Oxygen or O2 sensor circuitry will provide the ECU with the ability to more accurately adjust the fuel to air ratio which will in turn improve upon the overall performance of the engine. The O2 sensor works by measuring the oxygen content of the exhaust of the engine. Based on this measurement the fuel to air ratio of the engine is adjusted in order to optimize performance. There are three systems needed in order for the O2 sensor to be integrated into the ECU. The first is an internal resistance circuit. In order for the O2 sensor to work properly it needs to be heated to approximately 750 degrees Celsius which corresponds to an internal resistance of 80 ohms. Therefore a circuit is needed in order to measure the internal resistance of the sensor. This is done by using AC coupling to apply a higher frequency square wave to the pump cell and measuring the amount of displacement from the reference level voltage. Square wave levels above this level represent resistances above 80 ohms and square wave levels below the reference level represent resistances below 80 ohms.

The second circuit needed is a heater power circuit that uses the measure internal resistance of the sensor and applies a pulse width modulated signal with varying amplitude to a power MOSFET in order to maintain the proper heater temperature of the oxygen sensor. This circuit uses a MOSFET with its gate being driven by a pulse width modulated signal created by the microprocessor. The amount of time that the signal is high relates to the heater increasing temperature and vice versa the amount of time that the signal is low relates to the heater cooling.

The third and final circuit needed is the cell voltage circuit which is used to measure the actual fuel to air ratio of the exhaust. The cell voltage circuit is used to measure the Nernst cell voltage of the circuit using a differential amplifier with its output connected to a fixed resistor connected to the pump cell of the oxygen sensor. The cell voltage of the sensor is ideally 450mV which corresponds to a pump current of 0 amps and a lambda value of 1. The lambda value is the ratio of the measure fuel to air ratio to the stoichiometric fuel to air ratio. This ratio is calculated in the microprocessor based on the measured pump current and cell voltage. A lambda of greater than one, a cell voltage greater than 450mV, and a positive pump current relate to the engine running lean, meaning that there is too much oxygen mixing with the fuel. Conversely, a lambda less than one, a cell voltage less than 450 mV, and a negative pump current means that the engine is running rich, meaning that there is not enough oxygen mixing with the fuel. The ECU uses this data to adjust the injector and spark timing to improve engine performance. . Also for further information regarding the O2 sensor the following websites are helpful.

http://www.daytona-sensors.com/tech_wego.html
http://wbo2.com/lsu/lsuworks.htm

## PCB Layout Documentation for Rev 16

This document describes issues with the current PCB layout, and changes that should be made in the next revision. The design of the PCB started with the P08222 team using the PCB Artist software. Custom library files and all previous design files are available on EDGE. The design given to the P09222 team was not complete and had many issues, such as many connections that were not completed and parts that needed to be removed or changed. Due to the complexity of the PCB, the old design was kept but modified to meet the current design. Doing this allowed the board design to be completed more quickly at the expense of a quality design.

Rev 15 of the design was fabricated by Advanced Circuits, however many changes have been made in Rev 16 (the most current schematic design). All issues in this document that have been listed as fixed in Rev 16 are correct on the schematic, but not in the Rev 16 PCB layout file. PCB Artist uses a different file for the schematic and PCB. Schematic files end with .sch and PCB files end with .pcb. When working on the schematic/layout, save often and make backups. Backup the libraries as well at regular intervals. PCB Artist is very easy to use schematic and PCB design software. It can be downloaded from Advanced Circuits (www.4pcb.com). There are several tutorials available that show how to use the
major aspects of the software.

The following are issues with the current board:

• The schematic and the PCB layout were developed in parallel. This caused many changes to be made the the PCB layout which complicated routing. To make the most efficient routing, the PCB layout should only be completed after the final schematic is known to reduce unnecessary route changes.

• Layer 2 of the PCB is a ground plane but there is no power plane. Ideally, a power plane should be created to reduce noise on the signal lines. Signals should also be isolated from the power plane and other traces that carry high current, such as the injector outputs. To make the ground plane, make a rectangular shape that covers the entire board on layer 2. Then right click on the shape, and select “Pour Copper” to fill in the shape with copper. Uncheck “Thermals on Pads”, and “Thermals on Vias”. When the program fills in the shape, it will automatically remove the minimum amount of copper around vias that are not connected to ground. The design files for Rev 15 show the ground plane, while the shape was removed in Rev 16 to make editing the design easier.

• The 4 transistors driving the injectors should be placed close to the Motec connector to reduce the length of the high current traces and reduce noise to other parts of the board.

• There are two traces from the microcontroller to the Motec connector that will be used for a USB connection in the future. This is desired so that the ECU can be controlled and data read from a standard interface readily available to the Formula team. Currently on the board, the two USB traces seem to go randomly around the board to their destination. If a USB bus is implemented using the traces as is, it will almost certainly not work. Due to the high speed, differential nature of the USB bus, the two data lines need to be right next to each other for their entire length. The traces must also be *exactly* the same length. Investigate the USB spec as well as layout recommendations, for example, http://www.cypress.com/?rID=12982

• JTAG has been surfaced in the Motec connector so that the case will not have to be opened to reprogram the processor. The pins routed to the JTAG are scattered on the connector and not clustered together. Ideally, the pins for the JTAG should be clustered in the same physical section of the Motec connector to make the cabling harness simpler. Investigate modifying the pinout on the Motec connector to make this wiring easier.

• The Motec connector was too close to the edge of the PCB for it to fit properly in the case in Rev 15. This has been corrected in Rev 16.

• The 0.5A fuse on the 12V input (part FS1 on the schematic) will blow when powering on the board. It is suspected that this is due to an in-rush current, but needs to be verified.

• Several zener diodes were removed from the design in Rev 15 as they are unnecessary. The Fan/Fuel Pump circuit has been simplified.
• Be careful when placing components near the JTAG connector, as it is larger than the silkscreen outline. The silkscreen should be modified to reflect the actual size of the component.
• LED's could be added to the voltage regulators to provide a visual indication that the board is powered.
• Adding a reset button to the processor would help in debugging the board.
• Place pads for a zero-ohm resistor on all unused pins to the microcontroller if possible. This will allow easy access to large pads, allowing a wire to be soldered on should the pin be needed for later use or debugging purposes. Soldering wires onto 0.5mm pins is hard.
• The pin numbering on the Motec connector was backwards in Rev 15 and earlier. The connector has a total of 60 pins which is divided into two groups (A1 through A34 and B1 through 26). The documentation from the formula team uses the A and B pin labels to specify to which pin a signal is connected. In the old schematic, the part for the Motec connector had the pins labeled incorrectly, such that pin A1 was incorrectly labeled A34, A2 was incorrectly labeled A33, and so on. The B group had the same reversal in pin numbering. The schematic part has been corrected such that the labels are mapped to the correct physical pin, however ***the nets shown on the Rev 16 schematic have not yet been moved to the correct pin***. For example, according to the Formula team, the crank sensor needs to be on pin B1. On Rev 16 of the schematic, CrankIN is shown connected to pin B26. Rearrange the nets on the Rev 16 schematic such that CrankIN is connected to pin B1, CamIN connected to pin B2, etc. Use the datasheets from the Formula team to correct the schematic. Correcting the PCB layout may take a while to complete. To allow testing despite the incorrect pinout on the Motec connector, the test bench wiring has been altered such that the NIDAQ will work properly with the Rev 15 board.

## Rework Done to the Rev 15 Board:

These are changes that have been physically made to the Rev 15 board. These changes were necessary for the proper operation of the board, and are reflected in the Rev 16 schematic.
• The processor was missing the required connections to the voltage regulator to control the RESET pin. This has been fixed in Rev 16 and wires have been added to the PCB. These wires are very fragile and care must be taken in handling the board. Specifically, pin 18 of U8 is now connected to pin 32 of U25. R14 is now a 10k pull-up, and pin 5 and 19 of U8 are connected. Pin 6 of U8 is connected to +5V through a 0-ohm resister.
• The RST pin on the JTAG needed a 10k pull-up resister to function properly (R185).
• The PLL of the processor was not operating properly, and it was discovered that a connection was missing on the microcontroller. Thus, R148 was removed from the PCB and a wire was added connecting pin 73 (PLLDIS) to ground. • U3 was removed and the new injector circuit was added by connecting a perfboard to the PCB. This circuit was tested outside of the PCB on a breadboard, and should work correctly in the
circuit. Rev 16 of the schematic does NOT reflect the new injector design.

## Software

- Memory size of 30x15 table is 1.8kB
    - TMS470 RAM is 32kB
    - To set up all timings on cam, need:
        - 8x RPM estimate tables (1.8kB each)
            - Assumes RPM change while injectors are open has negligible effect on time injectors are open (~4ms)
            - If the variance is too high across range of RPMs over that time, need 12x tables, to estimate the end time.
- Interpolation with extrapolation results has not been implemented due to complexity
    - Since linear acceleration tables have varying $t_{n-1}$ granularity, interpolation becomes more complex.
- Ignition and injection fire at the same time for all. Since at each crank interrupt the timer is reset, and only injection and ignition for the current cylinder is set up, the other cylinders will fire based on previous settings. A workaround would be to clear all other timings each cycle.
- Better solution to tables for extrapolation is calculation, this way there is only one place where interpolation occurs and there is greater precision. Since the TMS470 is not recommended for new designs by TI, a new microcontroller should be specced out. Calculation time should be accounted for and the new uC should be fast enough for calculations to occur within timeframe.

- Changes to engine modeling equation may require change in processing power

## NIDAQ

### MAP, TPS, ECT, and IAT Sensor Models

Currently the MAP, TPS, ECT, and IAT sensors are inputs to the microprocessor generated by the NIDAQ Test Bench. However, these displays in the LabView vi's are 0-5V adjustable tabs and are set to arbitrary values during testing. Therefore, a model for each of these sensors should be created in order to accurately know what each sensor is measuring and to accurately set the sensor values for certain engine conditions, such as accelerating and decelerating. The datasheets for each sensor may explain the relationship between the voltage output by each sensor and the measured value by the sensor. Also the ECU software may also be useful in developing this relationship.

### TDMS File Viewing in Microsoft Excel

Both of the LabView vi's presently have a logging button on their respective front panels and when pressed the data being measured by LabView is written to a TDMS file. These files can be viewed in LabView using the TDMS viewer vi, but for further data manipulation it may be desirable to open the TDMS files in a program such as excel. An add-on for excel can be downloaded from the National Instruments website that is used to open TDMS files in excel. However, further investigation is needed as excel struggles with opening the TDMS files because they are larger than the number of cells available in excel.  Along with Excel, Matlab may also be an option to open the TDMS files for additional data viewing.

## O2 Sensor Signal Input

With the addition of the O2 sensor circuitry to the ECU and the future addition of the software needed to use the information from this sensor, the O2 sensor will need to be added to both of the LabView vi's.  This will allow for a more accurate depiction of the signals needed to properly model the operation of the engine and ECU.  For information regarding the O2 sensor refer to the O2 sensor section of this document and the Technical paper for this project.

## Acceleration and Deceleration on the Signal Quality vi

At this point in time the Timing vi is the only vi with the ability to do any form of acceleration or deceleration, therefore, it will be necessary to copy over the same programming that is need to do the acceleration and deceleration models from the Timing vi and integrate them into the signal quality vi.  Since the two vi's are very similar to one another it should just be a matter of copy and pasting the appropriate case structures from the Timing vi to the Signal Quality vi.

## Acceleration and Deceleration Models

At present the acceleration and deceleration models that have been integrated into the Timing vi are linear models.  This may not be accurate at all times during engine operation.  Therefore, it may be useful to create other types of models for specific engine conditions.  One way of doing this is to create an Excel spreadsheet containing RPM values that correspond to a desired engine condition to be modeled.  LabView contains write and read from spreadsheet sub-vi's that can be used to read in certain RPM values that are to be used at certain times.  By reading the RPM values from a spreadsheet the models for acceleration and deceleration can be greatly improved and the amount of calculations done by LabView can also be reduced.  In addition, currently the acceleration and deceleration models do not work when the measure button on the front panel is pressed.  This is because the update button during the acceleration and deceleration is set to true for the entire time the acceleration and deceleration models are running.  However, in order to update the measurements and allow for the measurements to be taken once the measure button is pressed, the update button needs to be toggled between true and false for each change in RPM.  This needs to be done in order to break the while loop that contains the measurement operation and update the RPM value.