

VI. FAULT-TOLERANT FUEL CONTROL SYSTEM

Summary The following example illustrates how to combine Stateflow with Simulink to efficiently model hybrid systems. This type of modeling is particularly useful for systems that have numerous possible operational modes based on discrete events. Traditional signal flow is handled in Simulink while changes in control configuration are implemented in Stateflow.

The model described below represents a fuel control system for a gasoline engine. The system is highly robust in that individual sensor failures are detected and the control system is dynamically reconfigured for uninterrupted operation.

Analysis and Physics Similar to the engine model described earlier in this document, physical and empirical relationships form the basis for the throttle and intake manifold dynamics of this model. The mass flow rate of air pumped from the intake manifold, divided by the fuel rate which is injected at the valves, gives the air-fuel ratio. The ideal, or stoichiometric mixture ratio provides a good compromise between power, fuel economy, and emissions. A target ratio of 14.6 is assumed in this system.

Typically, a sensor determines the amount of residual oxygen present in the exhaust gas (EGO). This gives a good indication of the mixture ratio and provides a feedback measurement for closed-loop control. If the sensor indicates a high oxygen level, the control law increases the fuel rate. When the sensor detects a fuel-rich mixture, corresponding to a very low level of residual oxygen, the controller decreases the fuel rate.

Modeling Figure 6.1 shows the top level of the Simulink model (fuel sys. mdl). The controller uses signals from the system's sensors to determine the fuel rate which gives a stoichiometric mixture. The fuel rate combines with the actual air flow in the engine gas dynamics model to determine the resulting mixture ratio as sensed at the exhaust.

The user can selectively disable each of the four sensors (throttle angle, speed, EGO and manifold absolute pressure [MAP]), to simulate failures. Simulink accomplishes this with Manual Switch blocks. Double-click on the block itself to change the position of the switch. Similarly, the user can induce the failure condition of a high engine speed by toggling the switch on the far left. A Repeating Table block provides the throttle angle input and periodically repeats the sequence of data specified in the mask.

Fault Tolerant Fuel Control System

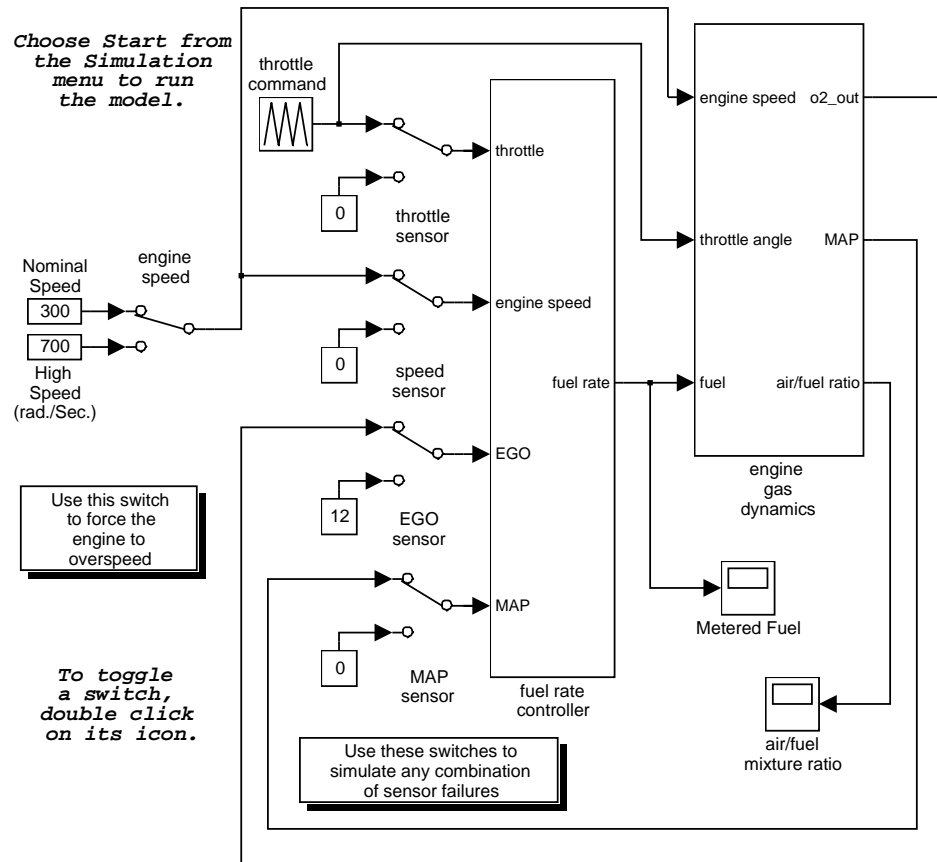


Figure 6.1: Simulink fuelsys model

The controller uses the sensor input and feedback signals to adjust the fuel rate to give a stoichiometric ratio (Figure 6.2). The model uses four subsystems to implement this strategy: control logic, sensor correction, airflow calculation, and fuel calculation. Under normal operation, the model estimates the airflow rate and multiplies the estimate by the reciprocal of the desired ratio to give the fuel rate. Feedback from the oxygen sensor provides a closed-loop adjustment of the rate estimation in order to maintain the ideal mixture ratio.

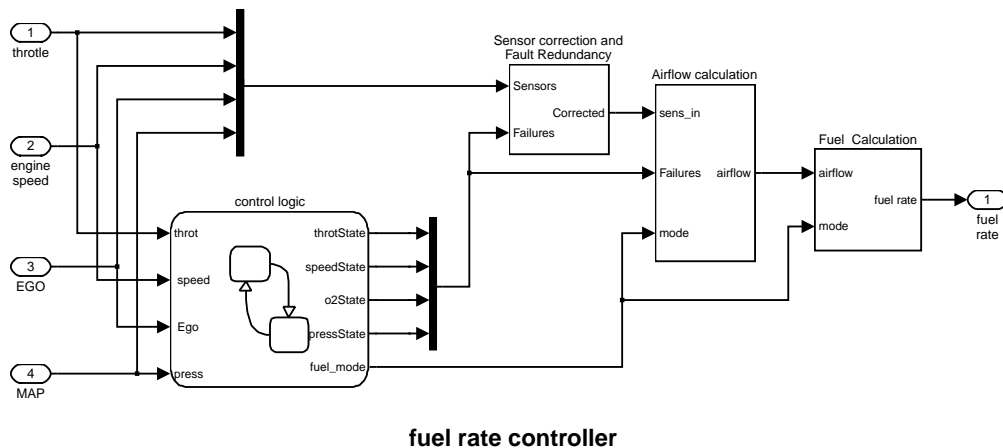


Figure 6.2: Fuel rate controller subsystem

Control Logic

A single Stateflow chart, consisting of a set of six parallel states, implements the control logic in its entirety. The four parallel states shown at the top of Figure 6.3 correspond to the four individual sensors. The remaining two parallel states at the bottom consider the status of the four sensors simultaneously and determine the overall system operating mode. The model synchronously calls the entire Stateflow diagram at a regular sample time interval of 0.01 sec. This permits the conditions for transitions to the correct mode to be tested on a timely basis.

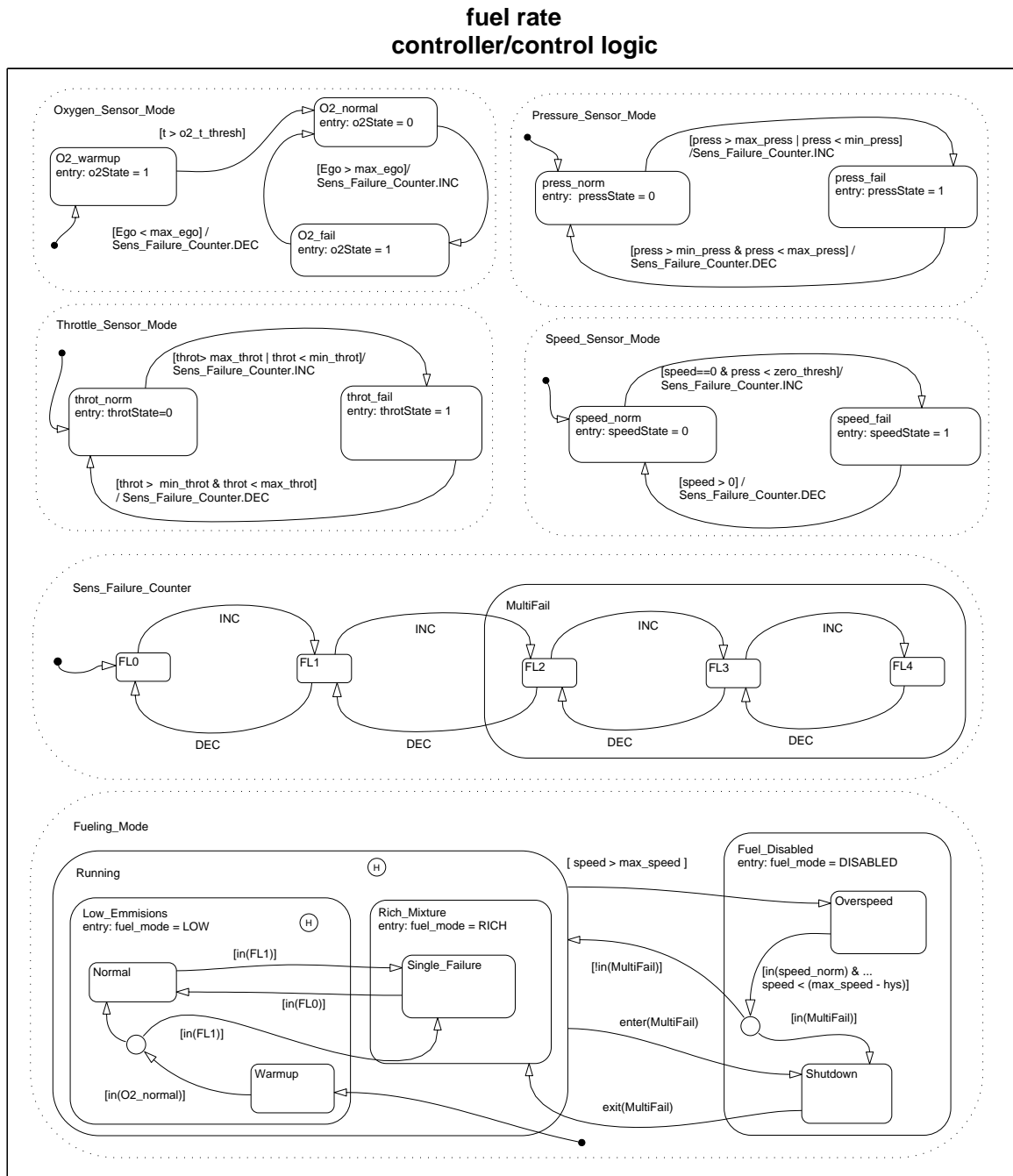


Figure 6.3: Control logic Stateflow diagram

When execution begins, all of the states start in their “normal” mode with the exception of the oxygen sensor. The O2_warmup state is entered initially until time has exceeded the *o2_t_thresh*. The system detects throttle and pressure sensor failures when their measured values fall outside their nominal ranges. A manifold vacuum in the absence of a speed signal indicates a speed sensor failure. The oxygen sensor also has a nominal range for failure conditions but, because zero is both the minimum signal level and the bottom of the range, failure can be detected only when it exceeds the upper limit.

Regardless of which sensor fails, the model always generates the directed event broadcast *Sens_Failure_Counter.INC*. In this way the triggering of the universal sensor failure logic is independent of the sensor. The model also uses a corresponding sensor recovery event, *Sens_Failure_Counter.DEC*. The *Sens_Failure_Counter* state keeps track of the number of failed sensors. The counter increments on each *Sens_Failure_Counter.INC* event and decrements on each *Sens_Failure_Counter.DEC* event. The model uses a superstate, *Multifail*, to group all cases where more than one sensor has failed.

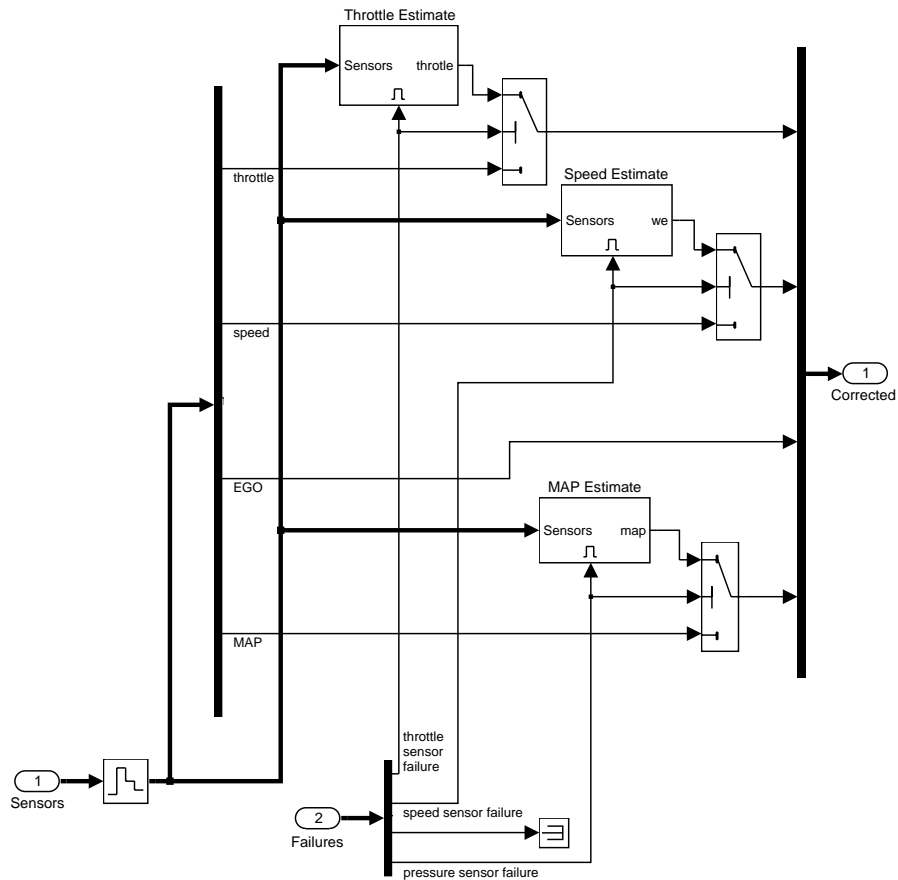
The bottom parallel state represents the fueling mode of the engine. If a single sensor fails, operation continues but the air/fuel mixture is richer to allow smoother running at the cost of higher emissions. If more than one sensor has failed, the engine shuts down as a safety measure, since the air/fuel ratio cannot be controlled reliably.

During the oxygen sensor warm-up, the model maintains the mixture at normal levels. If this is unsatisfactory, the user can change the design by moving the warm-up state to within the *Rich_Mixture* superstate. If a sensor failure occurs during the warm-up period, the *Single_Failure* state is entered after the warm-up time elapses. Otherwise, the *Normal* state is activated at this time.

We easily added a protective overspeed feature by creating a new state in the *Fuel_Disabled* superstate. Through the use of history junctions, we assured that the chart returns to the appropriate state when the model exits the overspeed state. As the safety requirements for the engine become better specified, we can add additional shutdown states to the *Fuel_Disabled* superstate.

Sensor Correction

The Fault Correction block determines which sensors to use and which to estimate. Figure 6.4 shows the block diagram for this subsystem. The failures input is a vector of logic signals that trigger the application of estimates to each particular sensor. When a component of the signal is nonzero, it enables the appropriate estimation subsystem and causes the switch relating to that signal to send the estimate as the output. Since the estimation routines are within enabled subsystems, they do not introduce any computational overhead when they are not needed.



6.4: Sensor correction and fault redundancy

The sensors input to the Correction block is the vector of raw sensor values. When there are no faults, the input simply passes on as the output signal. When a fault exists, the appropriate estimation block uses this signal to recover the missing component. Figure 6.5 shows an estimation example of the algorithm for the manifold pressure sensor.

MAP Estimation

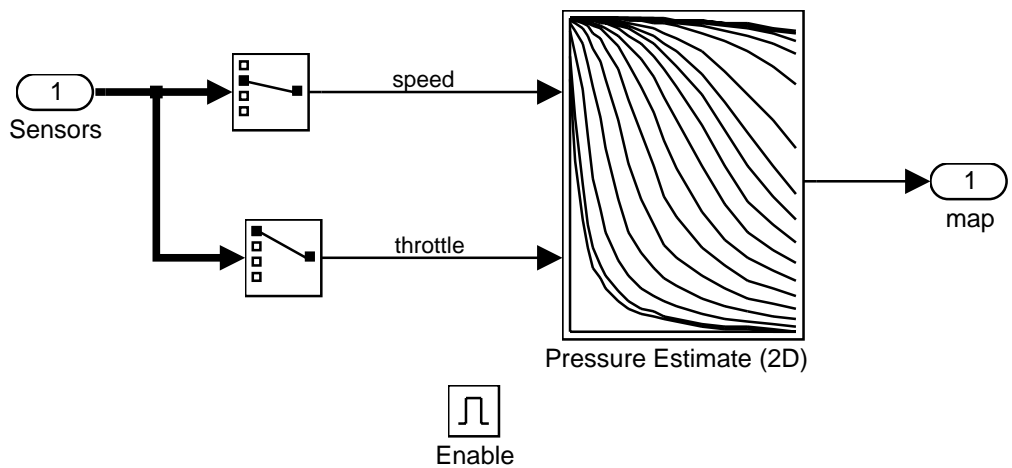


Figure 6.5: Manifold absolute pressure reconstruction

Airflow Calculation

The Airflow Calculation block (Figure 6.6) is the location for the central control laws. The block estimates the intake air flow to determine the fuel rate which gives the appropriate air/fuel ratio. Closed-loop control adjusts the estimation according to the residual oxygen feedback in order to maintain the mixture ratio precisely. Even when a sensor failure mandates open-loop operation, the most recent closed-loop adjustment is retained to best meet the control objectives.

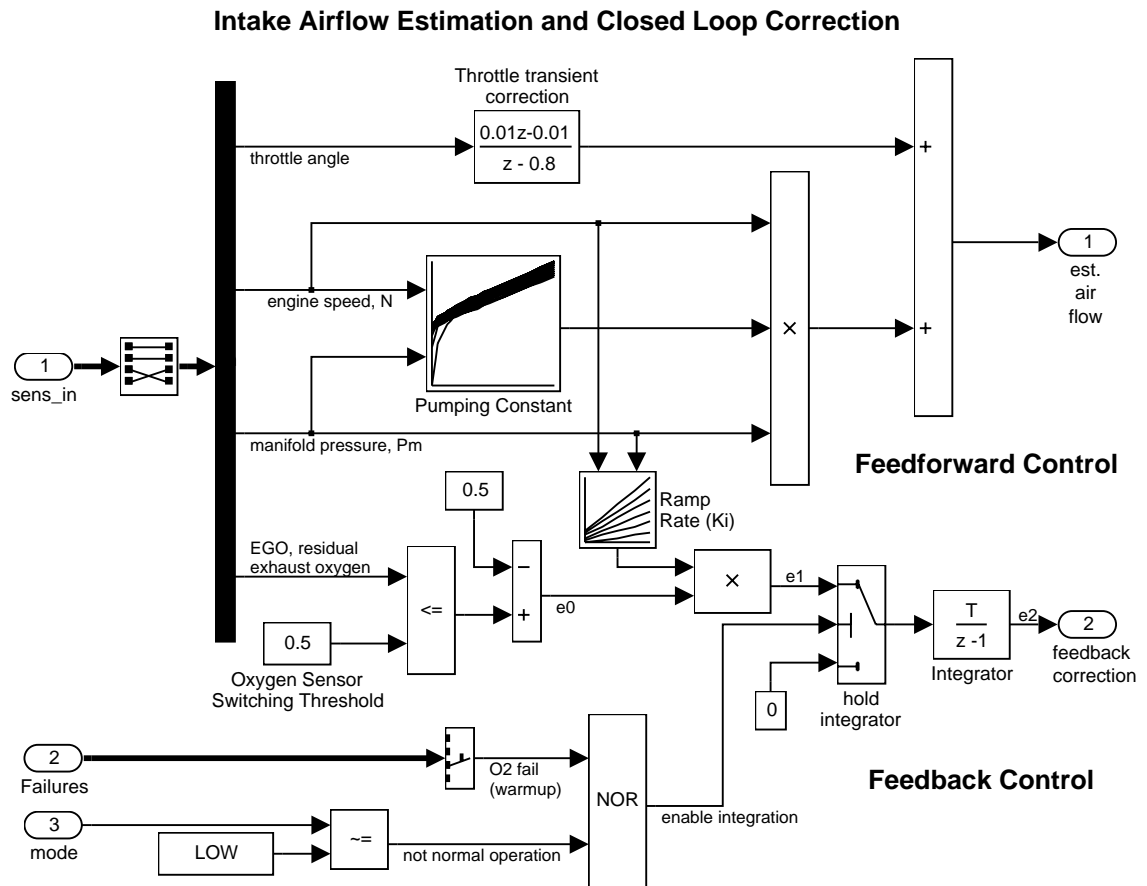


Figure 6.6: Airflow Estimation and Correction

The engine's intake air flow can be formulated as the product of the engine speed, the manifold pressure and a time-varying scale factor.

$$\begin{aligned}
 q &= \frac{N}{4\pi} V_{cd} \eta \frac{P_m}{RT} \\
 &= C_{pump}(N, P_m) N P_m, \\
 &= \text{intake mass flow}
 \end{aligned}$$

N = engine speed (rad/sec)

V_{cd} = engine cylinder displacement volume

Equation 6.1

η = volumetric efficiency

P_m = manifold pressure

R = specific gas constant

T = gas temperature

C_{pump} is computed by a lookup table and multiplied by the speed and pressure to form the initial flow estimate. During transients, the throttle rate, with the derivative approximated by a high-pass filter, corrects the air flow for filling dynamics. The control algorithm provides additional correction according to Eqs. 6.2:

$$e_0 = \begin{cases} 0.5, & \text{EGO} \leq 0.5 \\ -0.5, & \text{EGO} > 0.5 \end{cases}$$

$$e_1 = K_i(N, P_m)e_0$$

$$\dot{e}_2 = \begin{cases} e_1, & \text{LOW mode with valid EGO signal} \\ 0, & \text{RICH, DISABLE or EGO warmup} \end{cases}$$

Equation 6.2

e_2 = closed-loop correction

e_0, e_1, e_2 = intermediate error signals

The nonlinear oxygen sensor, modeled with a hyperbolic tangent in the engine gas Mixing and Combustion subsystem, provides a meaningful signal when in the vicinity of 0.5 volt. The raw error in the feedback loop is thus detected with a switching threshold, as indicated in Equation 6.2. If the value is low (the mixture is lean), the original air estimate is too small and needs to be increased. Conversely, when the oxygen sensor output is high, the air estimate is too large and needs to be decreased. Integral control is utilized so that the correction term achieves a level that brings about zero steady-state error in the mixture ratio.

The normal closed-loop operation mode, LOW, adjusts the integrator dynamically to minimize the error. The integration is performed in discrete time, with updates every 10 milliseconds. When operating open-loop however, in the RICH or O2 failure modes, the feedback error is ignored and the integrator is held. This gives the best correction based on the most recent valid feedback.

Fuel Calculation

The Fuel Calculation subsystem (Figure 6.7) sets the injector signal to match the given airflow calculation and fault status. The first input is the computed airflow estimation. This is multiplied with the target fuel/air ratio to get the commanded fuel rate. Normally the target is stoichiometric, 1/14.6.

When a sensor fault occurs, the Stateflow control logic sets the mode input to a value of 2 or 3 (RICH or DISABLED) so that the mixture is either slightly rich of stoichiometric or is shut down completely.

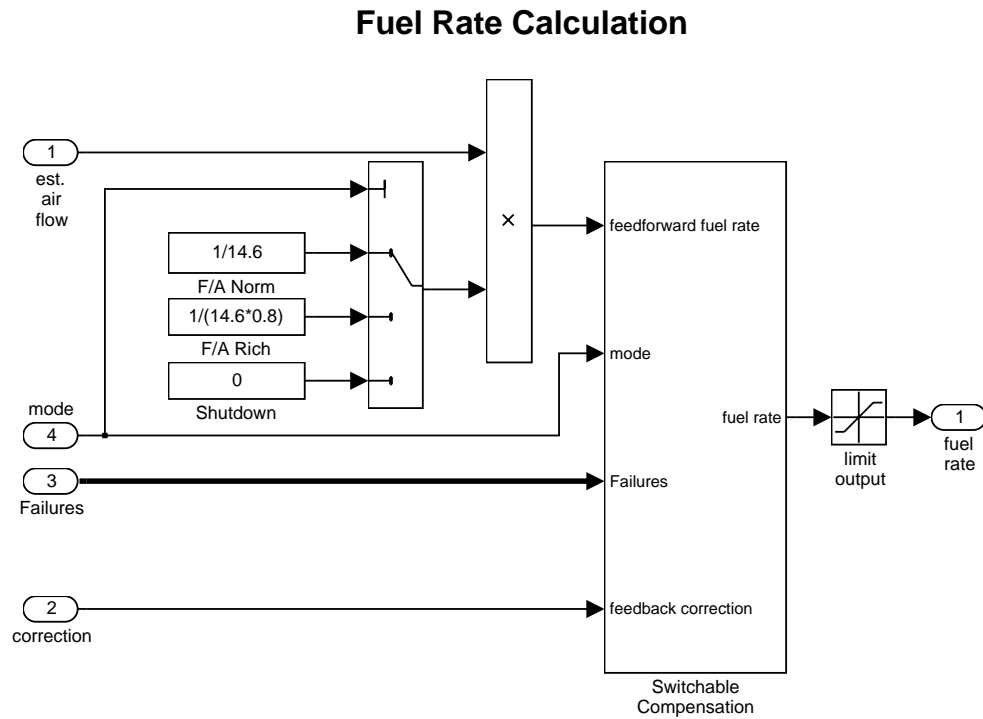
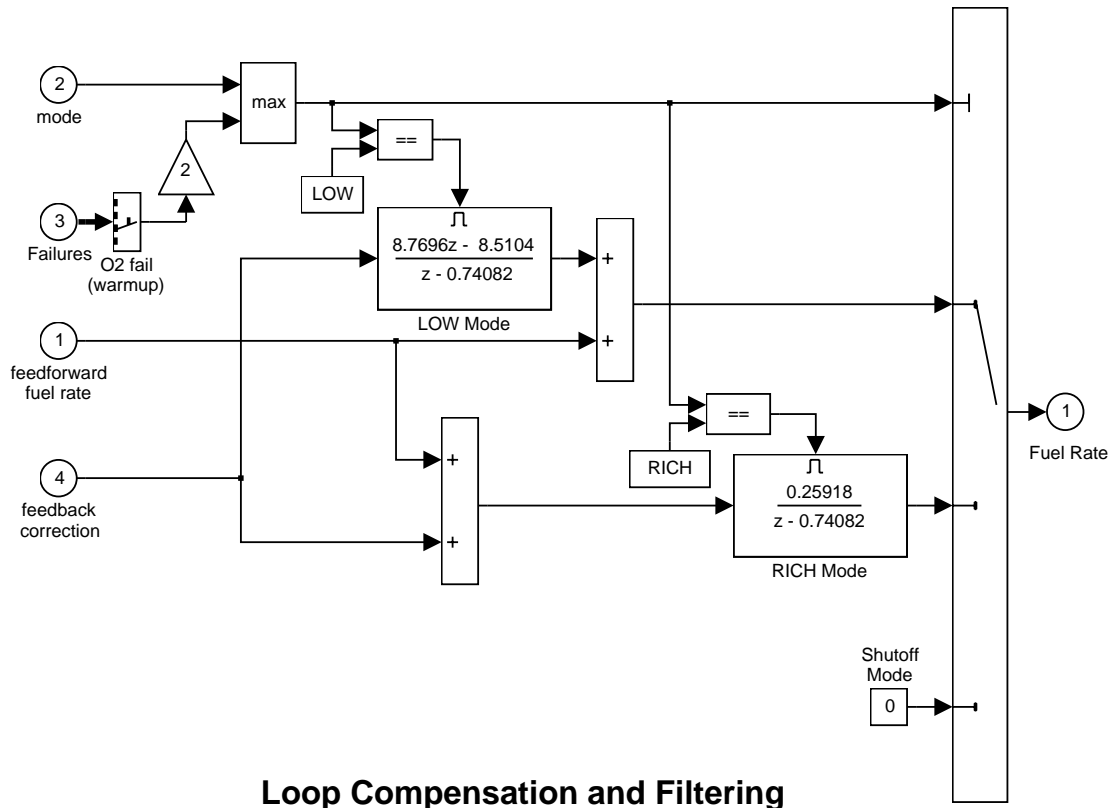


Figure 6.7: Fuel Calculation Subsystem

The Fuel Calculation subsystem (Figure 6.7) employs adjustable compensation (Figure 6.8) in order to achieve different purposes in different modes. In normal operation, phase lead compensation of the feedback correction signal adds to the closed-loop stability margin. In RICH mode and during EGO sensor failure (open loop), however, the composite fuel signal is low-pass filtered to attenuate noise introduced in the estimation process. The end result is a signal representing the fuel flow rate which, in an actual system, would be translated to injector pulse times.



Loop Compensation and Filtering

Figure 6.8: Switchable compensation

Results and Conclusions

Simulation results are shown in Figure 6.9 and Figure 6.10. The simulation is run with a throttle input that ramps from 10 to 20 degrees over a period of two seconds, then back to 10 degrees over the next two seconds. This cycle repeats continuously while the engine is held at a constant speed so that the user can experiment with different fault conditions and failure modes. To simulate a sensor failure, double-click on its associated switch (see Figure 6.1). Repeat this operation to toggle the switch back for normal operation.

Figure 6.9 compares the fuel flow rate under fault-free conditions (baseline) with the rate applied in the presence of a single failure in each sensor individually. Note, in each case, the nonlinear relationship between fuel flow and the triangular throttle command (shown qualitatively on its Simulink icon). In the baseline case, the fuel rate is regulated tightly, exhibiting a small ripple due to the switching nature of the EGO sensor's input circuitry. In the other four cases the system operates open loop. The control strategy is proven effective in maintaining the correct fuel profile in the single-failure mode. In each of the fault conditions, the fuel rate is essentially 125% of the baseline flow, fulfilling the design objective of 80% rich.

Figure 6.10 plots the corresponding air/fuel ratio for each case. The baseline plot shows the effects of closed-loop operation. The mixture ratio is regulated very tightly to the stoichiometric objective of 14.6. The rich mixture ratio is shown in the bottom four plots of Figure 6.10. Although they are not tightly regulated, as in the closed-loop case, they approximate the objective of $\text{air/fuel} = 0.8(14.6) = 11.7$.

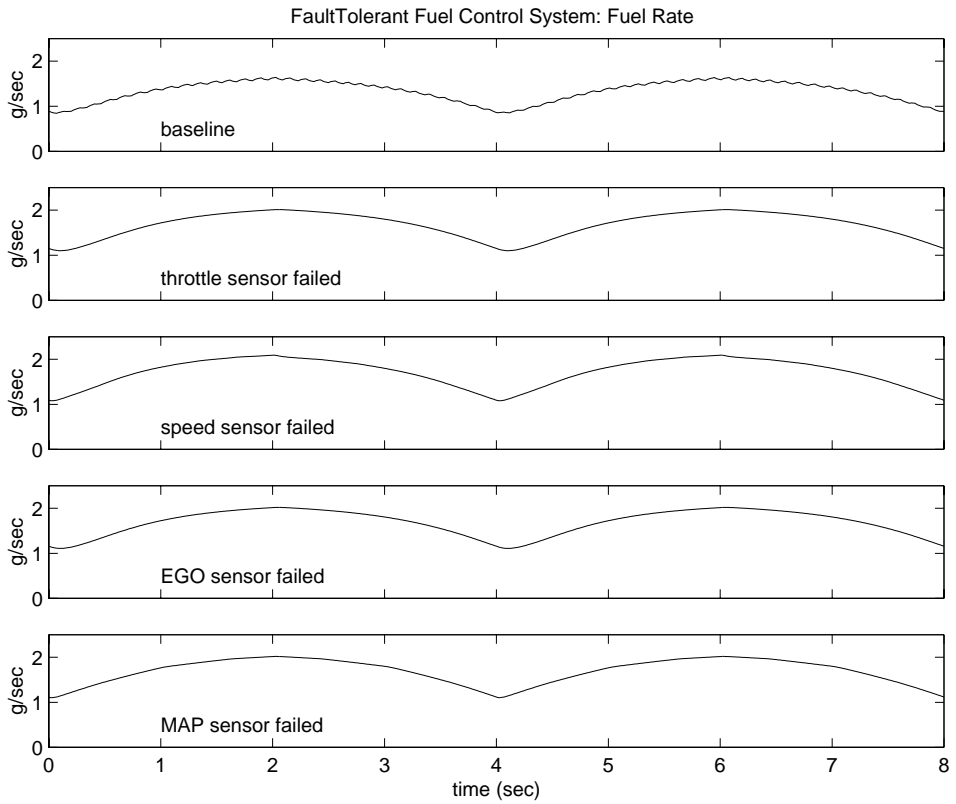


Figure 6.9: Comparative results for simulated fuel rate

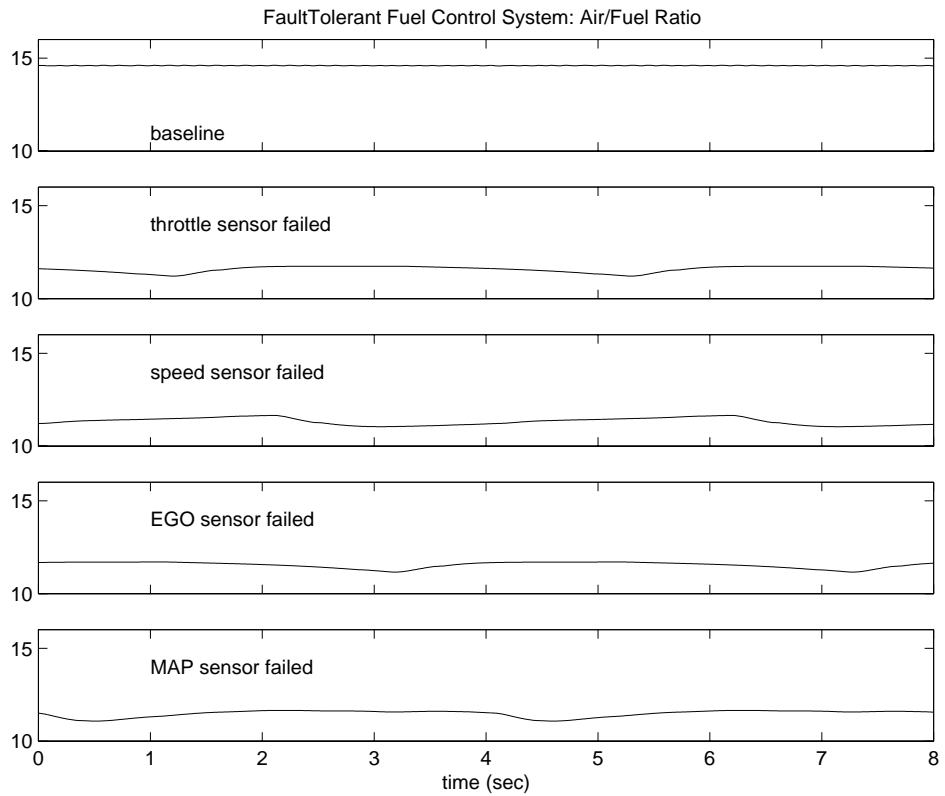


Figure 6.10: Comparative results for simulated air/fuel ratio

The transient behavior of the system is shown in Figure 6.11. With a constant 12° throttle angle and the system in steady-state, a throttle failure is introduced at $t = 2$ and corrected at $t = 5$. At the onset of the failure, the fuel rate increases immediately. The effects are seen at the exhaust as the rich ratio propagates through the system. The steady-state condition is then quickly recovered when closed-loop operation is restored.

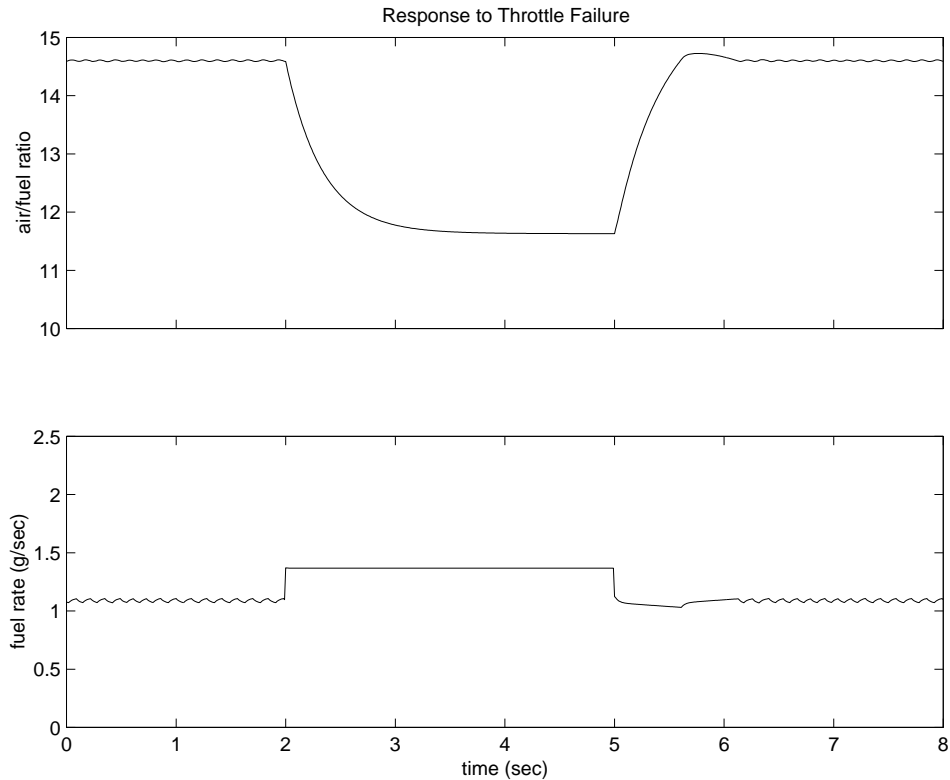


Figure 6.11: Transient response to fault detection

During simulation, this behavior can also be observed from the Stateflow perspective. By enabling animation in the Stateflow debugger, the state transitions are highlighted in Figure 6.3 as the various states are activated. The sequence of activation is indicated by changing colors. This closely coupled synergy between Stateflow and Simulink fosters the modeling and development of complete control systems. An engineer's concepts can develop in a natural and structured fashion with immediate visual feedback reinforcing each step.