



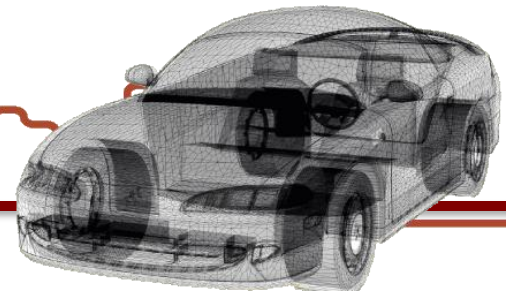
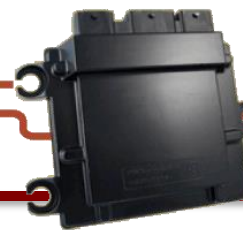
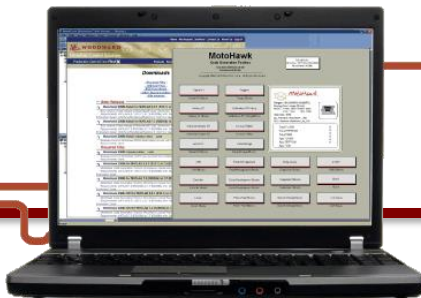
Raptor VeeCAN

Overview of Developing for the VeeCAN320
using the Raptor Platform



Feb. 26, 2014


Steve Sienkowski, Software Engineer
Dwayne McKenzie, Director of Distribution





Why Raptor VeeCAN?

- Customization
- Rapid prototyping
- Accessibility
 - Leverage application engineers familiar with Raptor or other model-based design tools to develop VeeCAN display applications


Solution Explorer

eventhandler.c
display.h
can.c
hard_h.h
public.h
public.c
PCInterface.cpp

(Global Scope)

```

    }
}

//-----
void process_rx_buffer(void)
{
    // Read all the messages from the CAN driver into our buffers
    int canPort;
    for (canPort = CAN_PORT1; canPort < CAN_MAX_PORTS; canPort++)
    {
        // Check for CAN messages received
        CAN_MSG_T canMsg;
        while (can_receive_message(canPort, &canMsg) == 0)
        {
            // Copy message to our buffer
            if(can_rx_head[canPort]++ == can_rx_end[canPort])
                can_rx_head[canPort] = can_rx_start[canPort];

            // ID
            can_rx_head[canPort]->arbitrator = canMsg.id << 3;

```

```

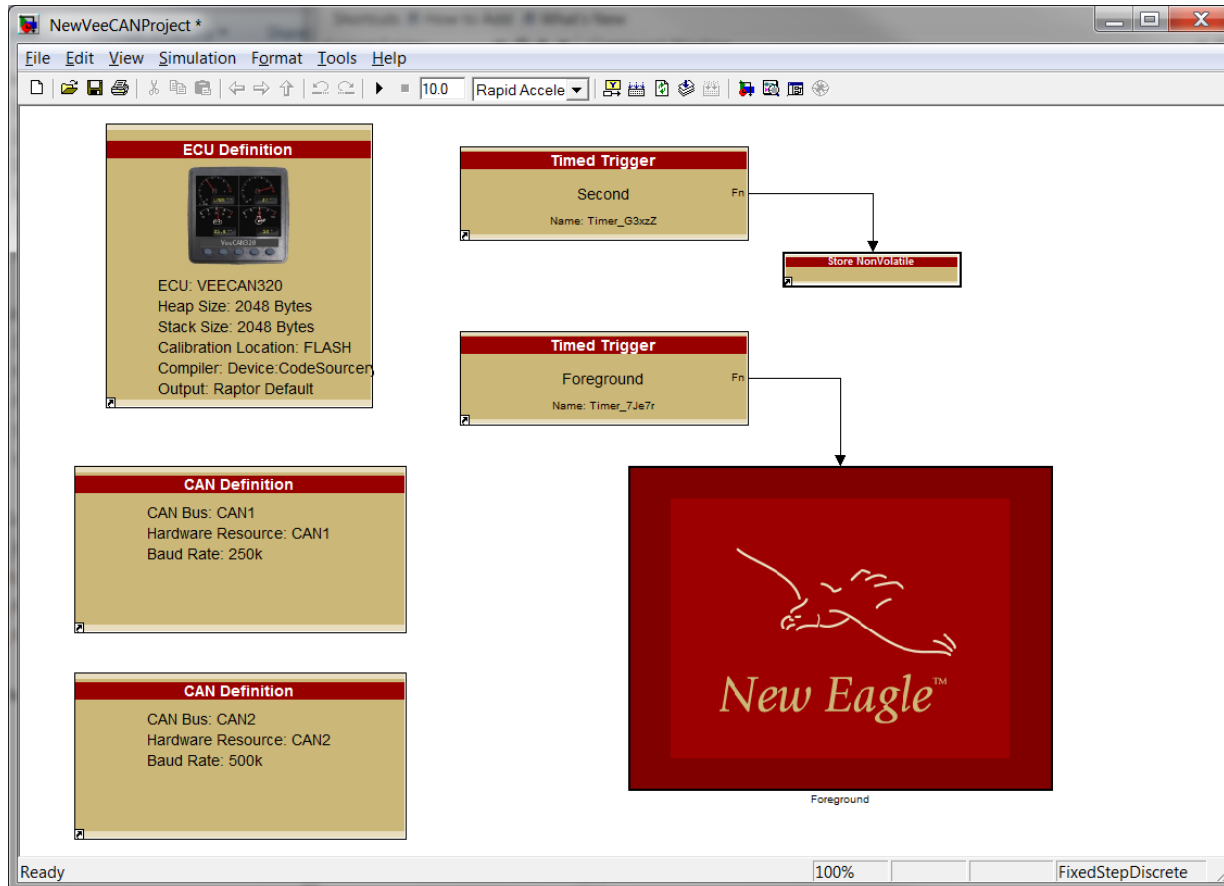
can_rx_head[canPort]->arbitrator = canMsg.id << 3;
\\ ID

```

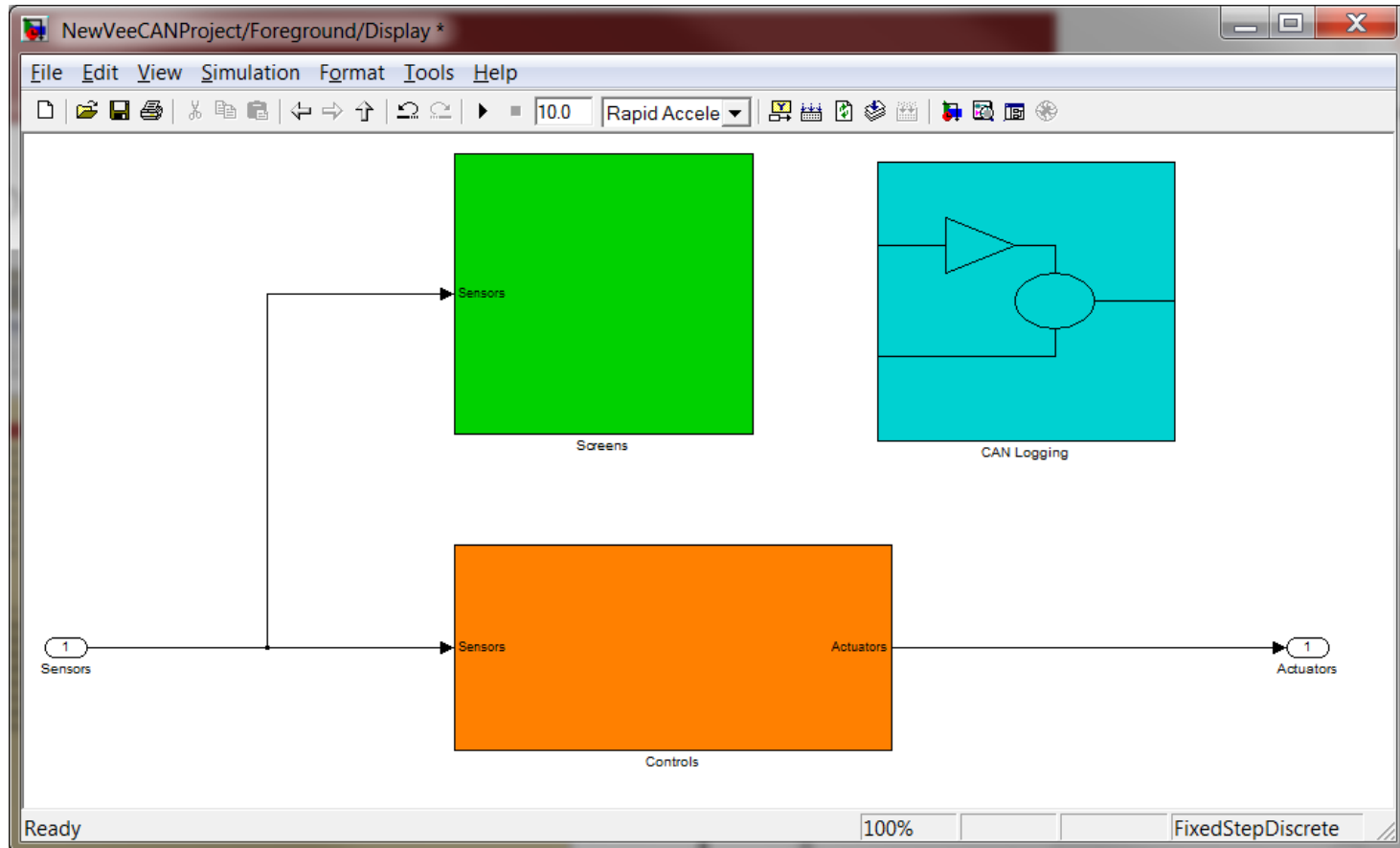
```

can_rx_head[canPort] = can_rx_start[canPort];
if(can_rx_head[canPort]++ == can_rx_end[canPort])
    // copy message to our buffer

```

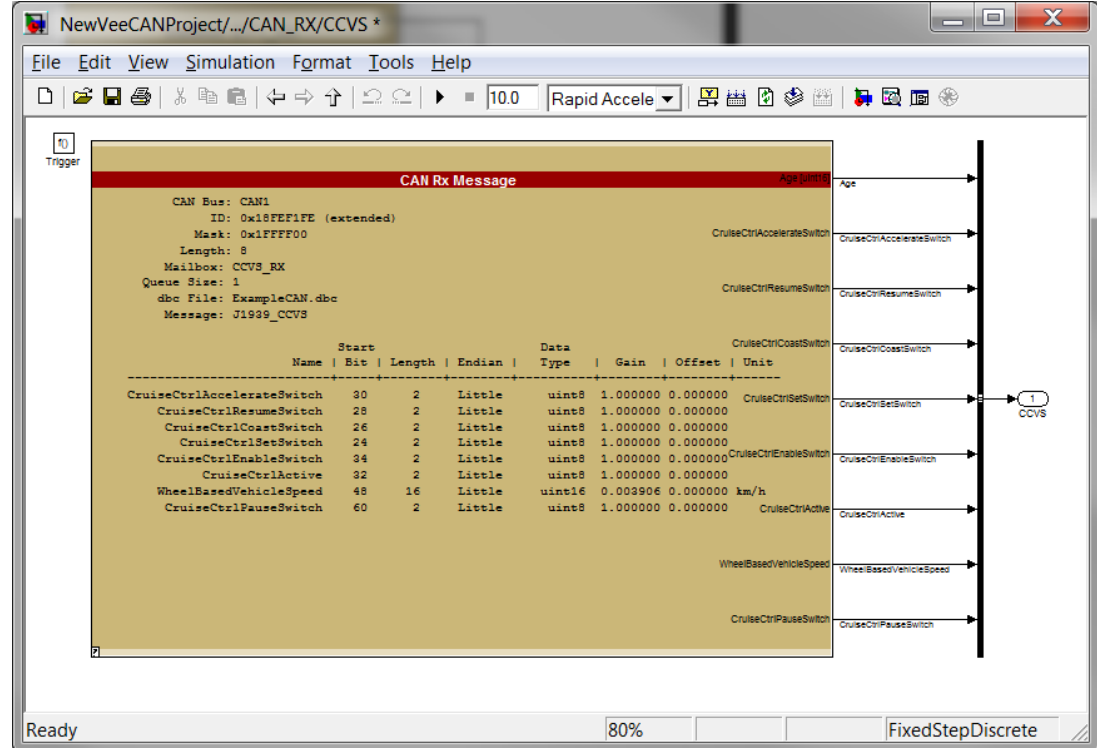
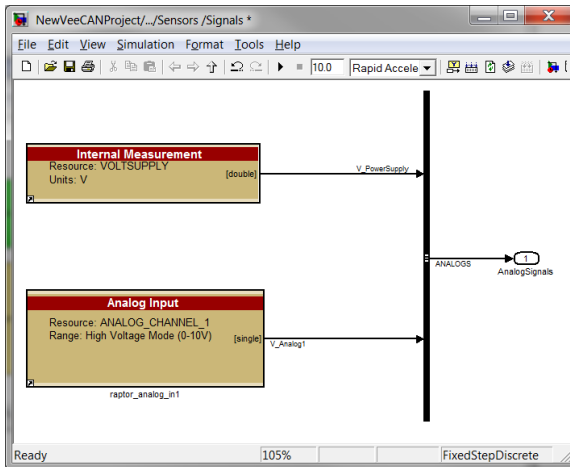


Model-based Control

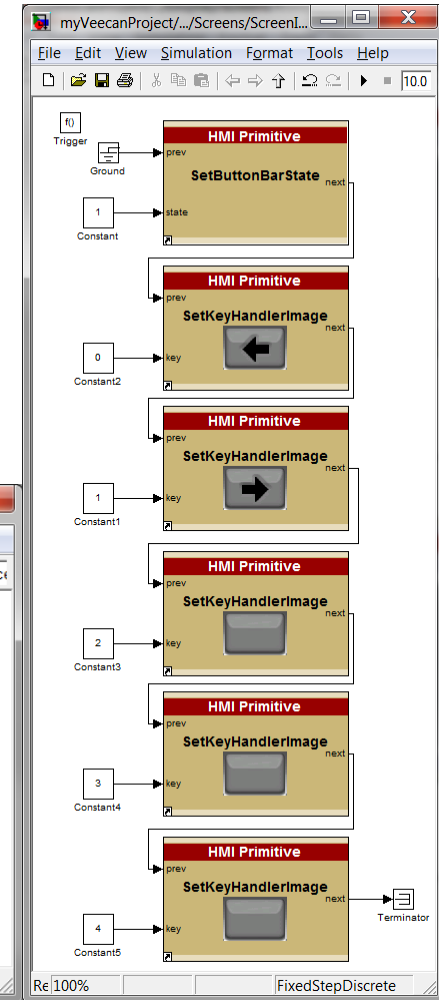
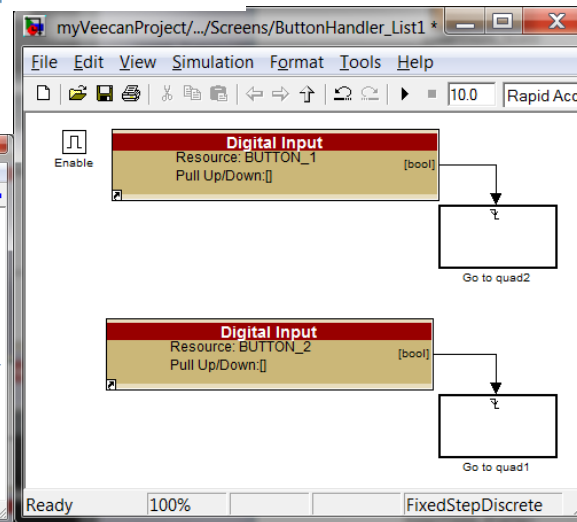
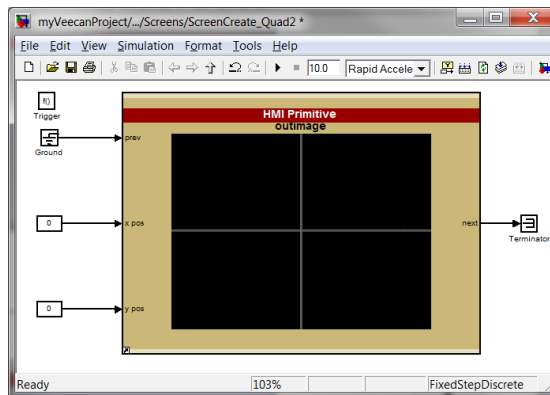
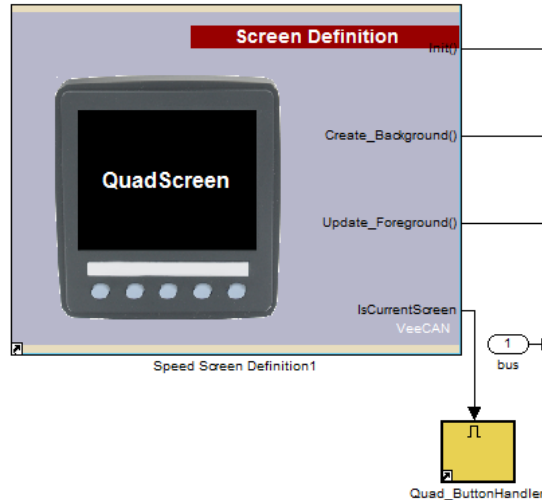




Inputs (Analog, CAN)

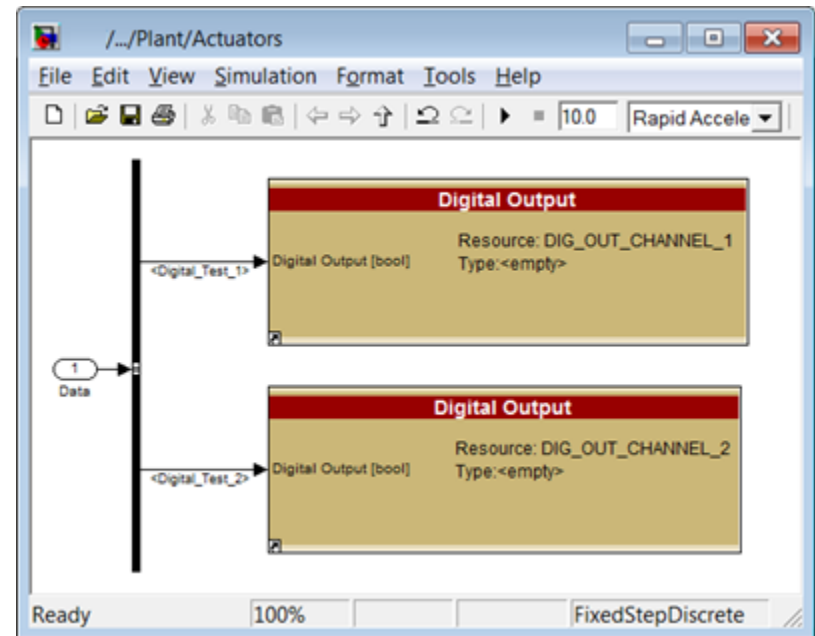
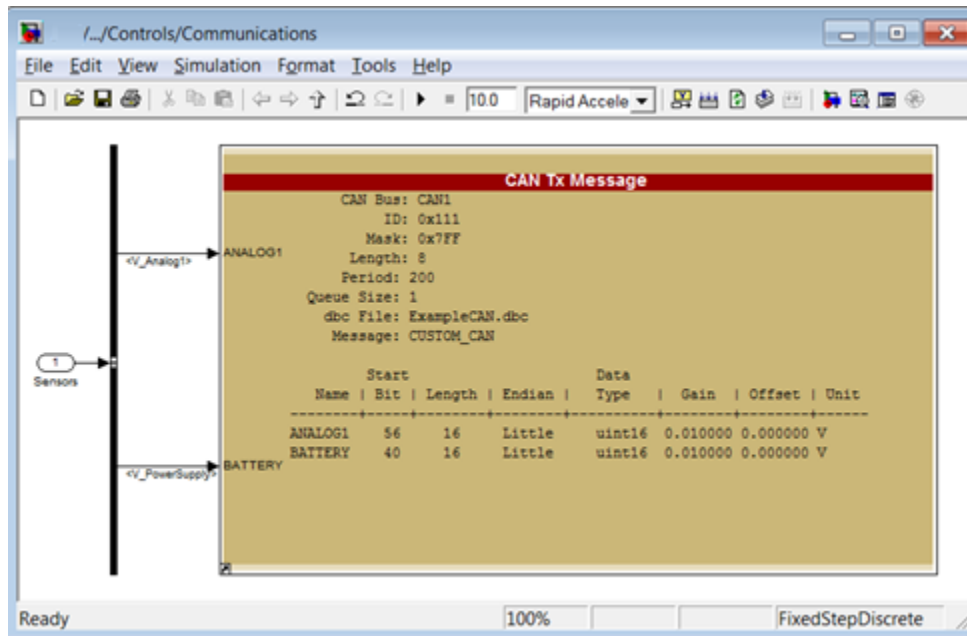


Screen Development



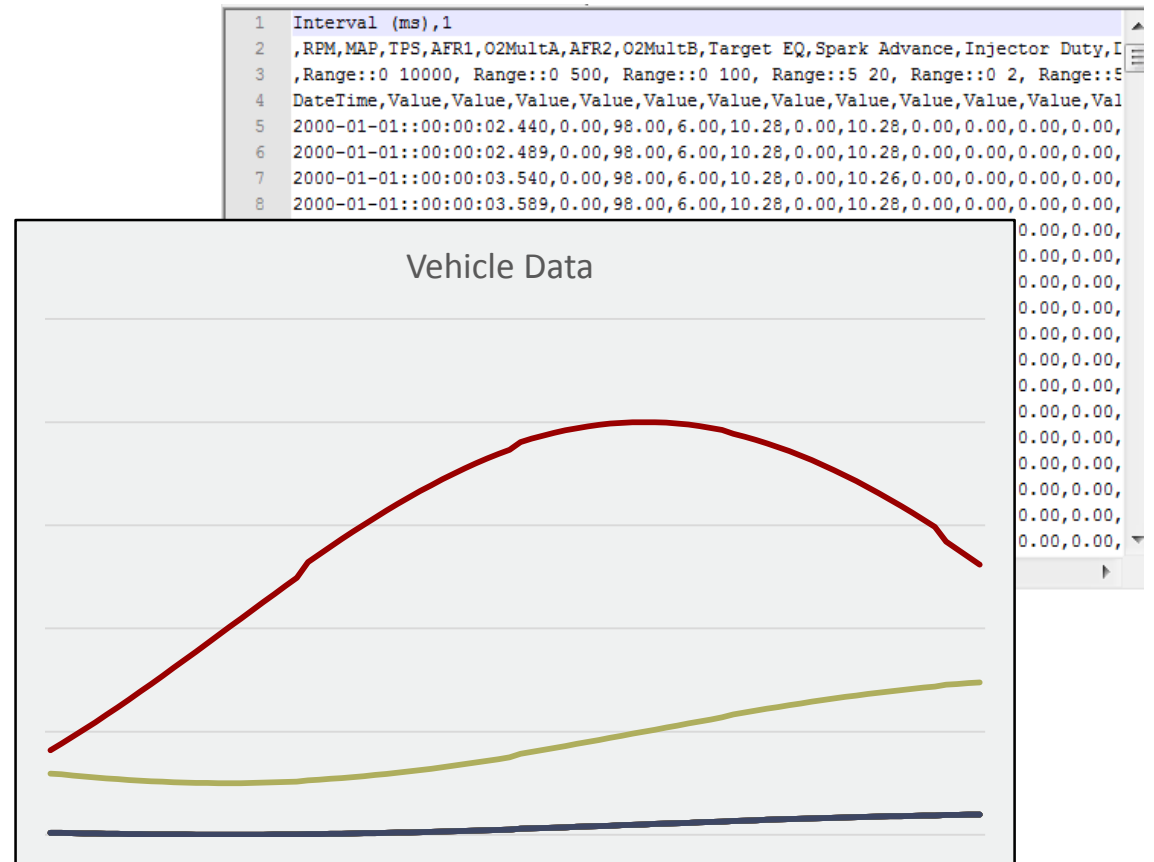
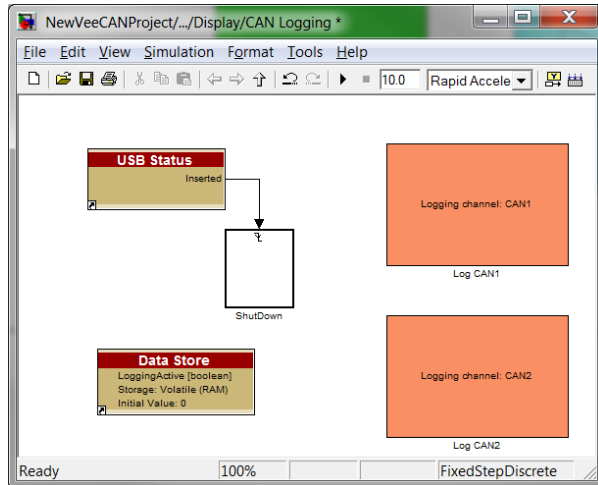


Outputs (CAN, Digital)





CAN Logging



Building, Simulating

