V2X Motorcycle HUD Weekly Updates

•••

Ryan Hiser, Jacob Nguyen, Jorge Pacheco
UCSD MAS WES
12 March 2022

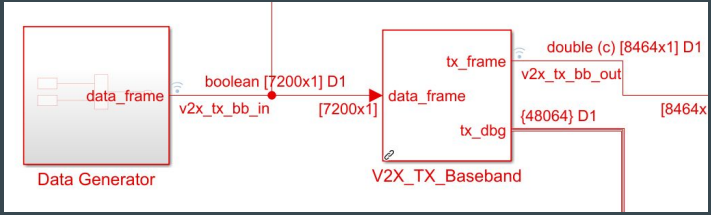
Planned Progress (Previous Two Weeks)

- Kastner approval (Still Waiting...)
- Fix FMcomms 4 (Complete!!)
- HUD server/client software and hotspot (Complete!!)
- Waveform improvements (On-going!!)

Embedded Coder

MATLAB/Simulink Updates

- Logs data for unit test
- Generates source code




```
41  % V2X TX Baseband
42  if save_to_csv
43      v2x_tx_bb_in  = squeeze(sim_out.logsout.getElement('v2x_tx_bb_in').Values.Data);
44      v2x_tx_bb_out = squeeze(sim_out.logsout.getElement('v2x_tx_bb_out').Values.Data);
45      writematrix(v2x_tx_bb_in,  'main/data/v2x_tx_bb_in.csv')
46      writematrix(real(v2x_tx_bb_out), 'main/data/v2x_tx_bb_out_real.csv')
47      writematrix(imag(v2x_tx_bb_out), 'main/data/v2x_tx_bb_out_imag.csv')
48  end
49
50  %% Build script
51  if 1
52      % Models
53      v2x_tx_bb_fp  = 'v2x_modem_tb/V2X_TX_Baseband';
54      v2x_tx_mod_fp = 'v2x_modem_tb/V2X_TX_Modulator';
55      v2x_rx_bb_fp  = 'v2x_modem_tb/V2X_RX_Baseband';
56
57      % Generated code folder
58      src_fp = 'src';
59      if isfolder(src_fp)
60          rmdir(src_fp, 's');
61      end
62      if ~isfolder(src_fp)
63         mkdir(src_fp)
64      end
65      set_param(0, 'CodegenFolder', src_fp)
66
67      % V2X TX Baseband
68      slbuild(v2x_tx_bb_fp)
69      slbuild(v2x_tx_mod_fp)
70      slbuild(v2x_rx_bb_fp)
71  end
```

Signals saved to CSV

33	0	0	0	1
34	0	0	0	0
35	0	0	1	1
36	0	0	0	0
37	0	1	0	0
38	1	0	0	1
39	1	0	0	0
40	1	1	1	1
41	1	0	0	0
42	0	1	1	0
43	0	0	1	1
44	0	1	1	0
45	1	1	1	1
46	0	1	1	0
47	0	0	0	1
48	1	1	0	0
49	1	0	0	0
50	1	1	1	1
51	0	0	0	1
52	1	1	0	1
53	1	0	1	0
54	1	1	1	0
55	0	0	0	0
56	0	0	1	0

v2x_tx_bb_in

Generated Code

V2X_TX_Baseband.c | V2X_TX_Baseband.h | V2X_TX_Baseband_data.c

```
1   /*
2    * Academic License - for use in teaching, academic research, and meeting
3    * course requirements at degree granting institutions only.  Not for
4    * government, commercial, or other organizational use.
5    *
6    * File: V2X_TX_Baseband.c
7    *
8    * Code generated for Simulink model 'V2X_TX_Baseband'.
9    *
10   * Model version                  : 1.134
11   * Simulink Coder version         : 9.6 (R2021b) 14-May-2021
12   * C/C++ source code generated on : Sun Mar  6 18:32:02 2022
13   *
14   * Target selection: ert.tlc
15   * Embedded hardware selection: Intel->x86-64 (Linux 64)
16   * Code generation objectives:
17   *    1. Execution efficiency
18   *    2. Traceability
19   * Validation result: Not run
20   */
21
22  #include "V2X_TX_Baseband.h"
23
24  /*===========*
25   * Constants *
26   *===========*/
27  #define RT_PI                          3.14159265358979323846
28  #define RT_PIF                         3.1415927F
29  #define RT_LN_10                       2.3025850929940568402
30  #define RT_LN_10F                      2.3025851F
31  #define RT_LOG10E                      0.43429448190325182765
32  #define RT_LOG10EF                     0.43429449F
33  #define RT_E                           2.7182818284590452354
34  #define RT_EF                          2.7182817F
```

Embedded Coder

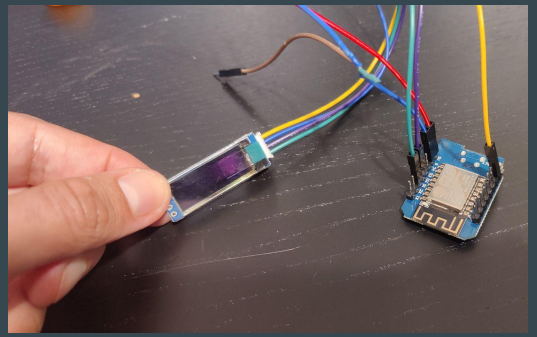
- Generated Code is C
- Example testbench created to verify generated code is working as intended
- Testbench passes! Generated code output matches CSV values
- Next steps:
  - Create robust SW model/diagram
  - Create periodic (every 0.01s) TX/RX service
  - Integrate with motorcycle status updates/HUD

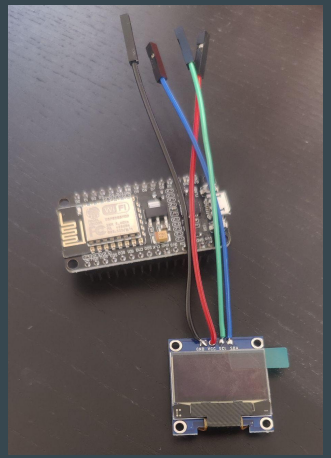
```
157  // ---------------------- Main ----------------------
158  int_T main(int_T argc, const char *argv[])
159  {
160      RT_MODEL *const rtM = rtMPtr;
161
162      /* Unused arguments */
163      (void)(argc);
164      (void)(argv);
165
166      /* Pack model data into RTM */
167      rtM->dwork = &rtDW;
168
169      /* Initialize model */
170      V2X_TX_Baseband_initialize(rtM, rtU_data_frame, rtY_tx_frame, rtY_tx_in,
171      rtY_scrambler_out, rtY_encoder_out, rtY_mapper_out, rtY_preamble_out);
172
173      // Set mumber of frames to compare
174      int num_frames = 4;
175      for (int i = 1; i <= num_frames; i++)
176      {
177          loadCSV(i);
178          rt_OneStep(rtM);
179          int ret_val = compareOut();
180
181          if (ret_val != 0)
182          {
183              printf("Frame %d does NOT match recorded CSV!\n", i);
184          }
185          else
186          {
187              printf("Frame %d matches recorded CSV!\n", i);
188          }
189      }
190
191      return 0;
192  }
```

```
jnguyen@BLD:/mnt/c/Users/jacob/Documents/Important/UCSD/2021-2022 Year/WES Capstone/capstone/models/main$ ./v2x_tx_bb_test
Frame 1 matches recorded CSV!
Frame 2 matches recorded CSV!
Frame 3 matches recorded CSV!
Frame 4 matches recorded CSV!
```

HUD

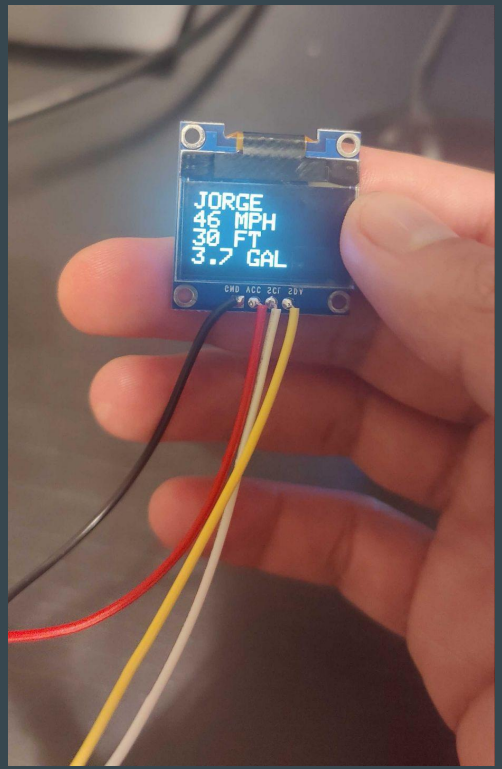
- Encountered issues with first batch of components
  - Assembled all parts and soldered components
  - Created test programs on Arduino IDE to configure ESP8266 (WiFi) and SSD1306 (OLED)
  - Couldn't control OLED with SPI or I2C.
  - Ordered a new set or parts with same controllers for both OLED and WiFi module





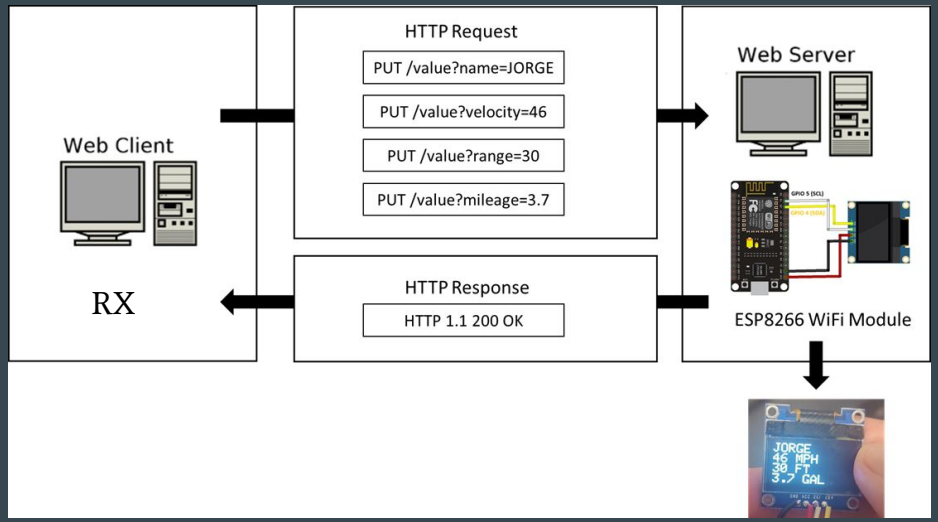
HUD (cont)

- Configured ESP8266 WiFi module + OLED to receive subset of data we plan to transmit
  - Created test programs to fiddle with WiFi server functionality
  - Started with Arduino IDE and commanded OLED screen directly
  - Created simple server on ESP8266 and connected it to local network
  - Tested capabilities of OLED screen and learned how to control positioning, text sizes, etc



HUD (cont)

- Configured ESP8266 to create is own WiFi network
- Configured ESP8266 as a simple HTTP server
- Clients can update any relevant values through simple HTTP commands:

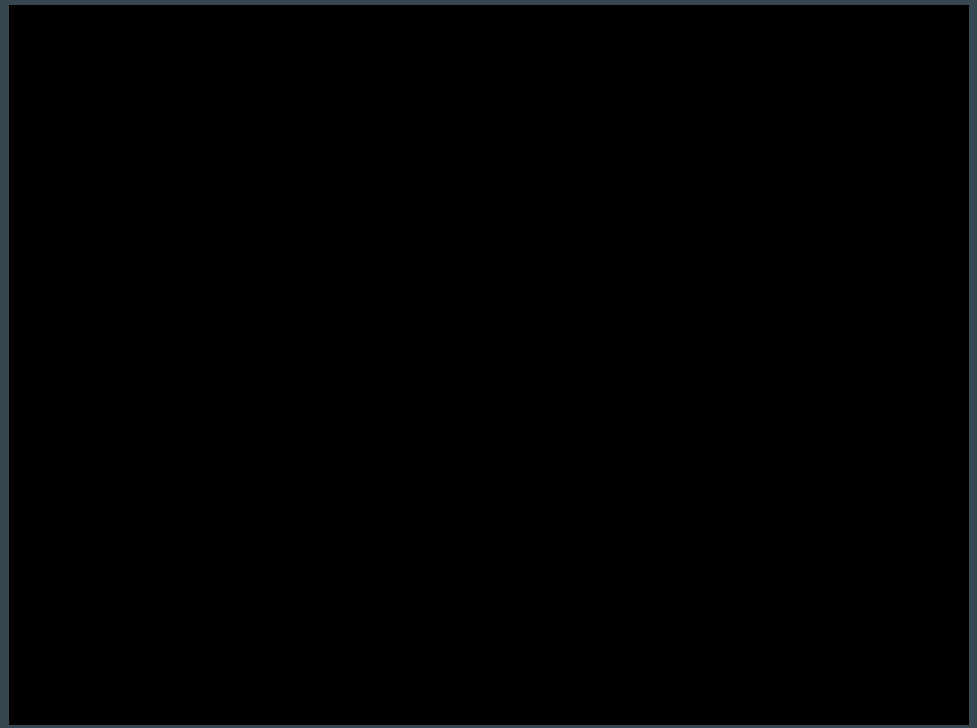


- Next steps
  - Work on enclosure and integrate with system

Zed + FMComms4

- Successfully used ADI image with meta-adi yocto layer and petalinux generated image.

- Video demonstrates a transmitted sinusoid feedback to the ADC via a SMA cable.
  - Attenuation set high to ensure safety, due to uncertainty of signal levels.



Video shows transmitted and received signal. As well as Zed board with SMA loop back (bottom left).

Fmcomms + Pynq

- Pynq image works with libiio
  - Created a program to read (from ADC) and write (to DAC).

- Next steps:
  - Need an attenuator to safely test.
  - Also need to investigate libiio registers (internal attenuation).
  - Then can integrate with modulator C/C++ code.
  - Duplicate SD card image.

FMCOMSS Loop Back Test RX -> TX

In [2]:
```
TXLO = 400e6
TX0BW = 2.5e6
TX0FS = 2.5e6
RXLO = 500e6
RX0BW = 2.5e6
RX0FS = 2.5e6
```

In [3]:
```
# Setup IIO Context and device handles
ctx = iio.Context()
ctrl = ctx.find_device("ad9361-phy") # Register control
txdac = ctx.find_device("cf-ad9361-dds-core-lpc") # TX/DAC Core in HDL for DMA (plus DDS)
rxadc = ctx.find_device("cf-ad9361-lpc") # RX/ADC Core in HDL for DMA
```

In [4]:
```
# Set LO, BW, FS for TX/RX
ctrl.channels[1].attrs["frequency"].value = str(int(TXLO))
ctrl.channels[5].attrs["rf_bandwidth"].value = str(int(TX0BW))
ctrl.channels[5].attrs["sampling_frequency"].value = str(int(TX0FS))
ctrl.channels[0].attrs["frequency"].value = str(int(RXLO))
ctrl.channels[4].attrs["rf_bandwidth"].value = str(int(RX0BW))
ctrl.channels[4].attrs["sampling_frequency"].value = str(int(RX0FS))
```

In [5]:
```
# Enable I/Q channels to be associated with RX buffer
rxadc.channels[0].enabled = True
rxadc.channels[1].enabled = True
txdac.channels[4].enabled = True
txdac.channels[5].enabled = True
```

In [6]:
```
# Create IIO Buffers
rxbuf = iio.Buffer(rxadc, 8192, False) # False = non-cyclic buffer
txbuf = iio.Buffer(txdac, 8192, False) # False = non-cyclic buffer
```

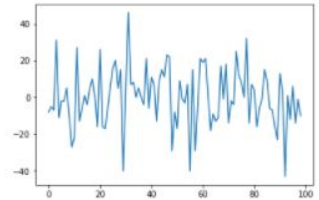
Send and Receive (Read from ADC, write to DAC)

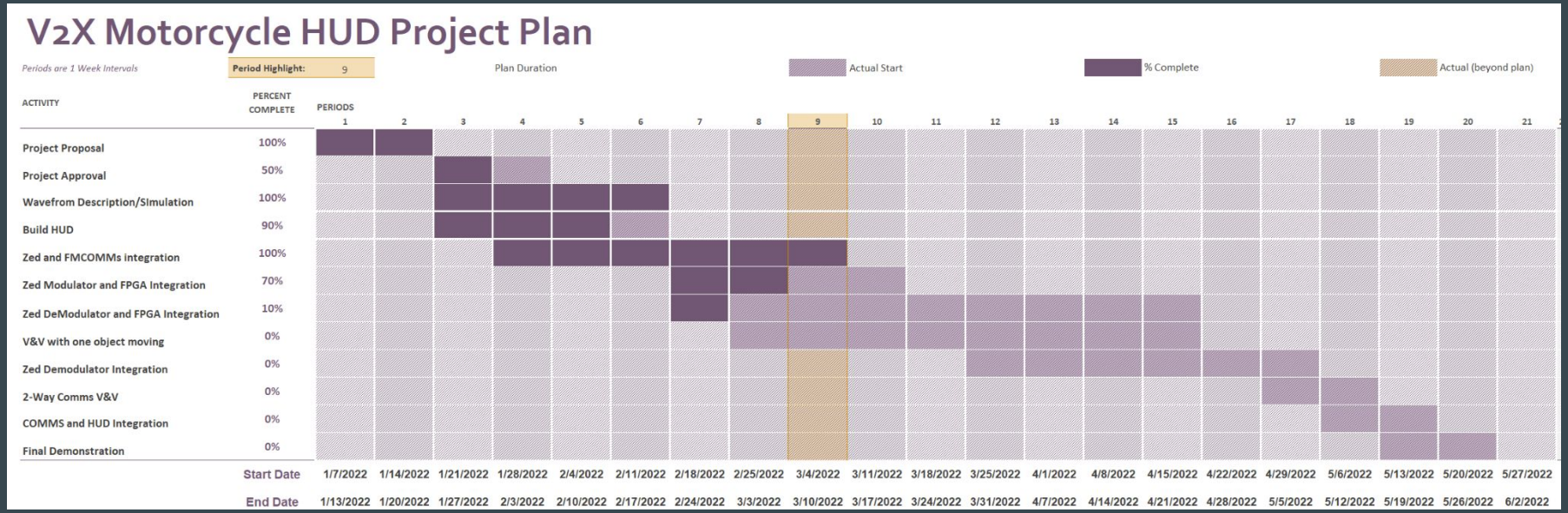
In [7]:
```
# perform test
for i in range(4000):
    rxbuf.refill()
    x = rxbuf.read()
    txbuf.write(x)
    txbuf.push()
```

In [8]:
```
import numpy as np
y = np.frombuffer(x,dtype='int16')
```

Below is random noise because I do not have an atennutor and could not find the right values to set the TX attenuation.

In [10]:
```
import matplotlib.pyplot as plt
plt.plot(y[1:100])
plt.show()
```





Planned Progress (Next Two Weeks)

- Kastner approval
- Integrate C-code modulator with FMCOMMS4.
- Complete HUD mechanical.
- Continue work on Timing Error Recovery (FPGA design).