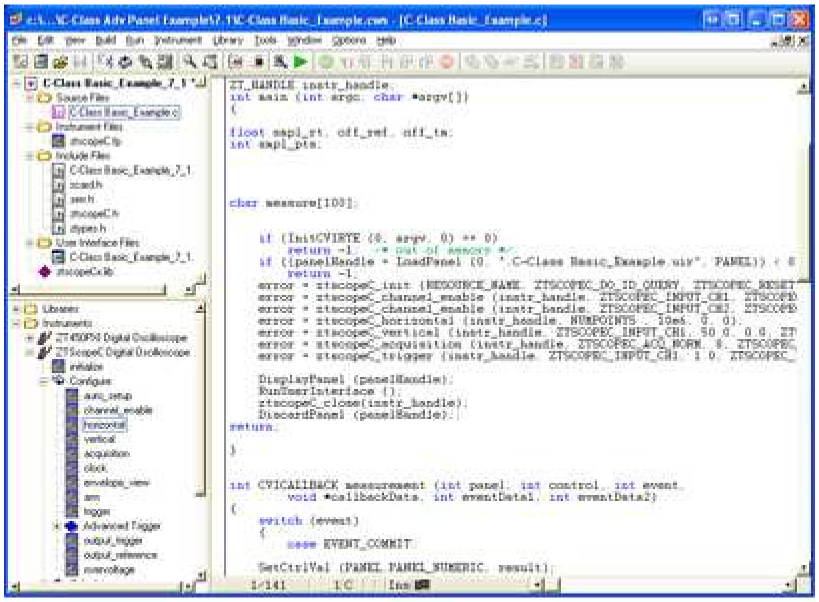
# DiagnoSYS Subway Electronics Functional Test Solution Using PXI

DiagnoSYS has incorporated the  PXI modular digitizers into a functional test solution for subway car electronics validation. DiagnoSYS is a global company that leverages industry standard hardware and software to create flexible and scalable turnkey test solutions.  PXI's flexible software drivers and powerful built-in measurements proved to be a great match for DiagnoSYS's test system.

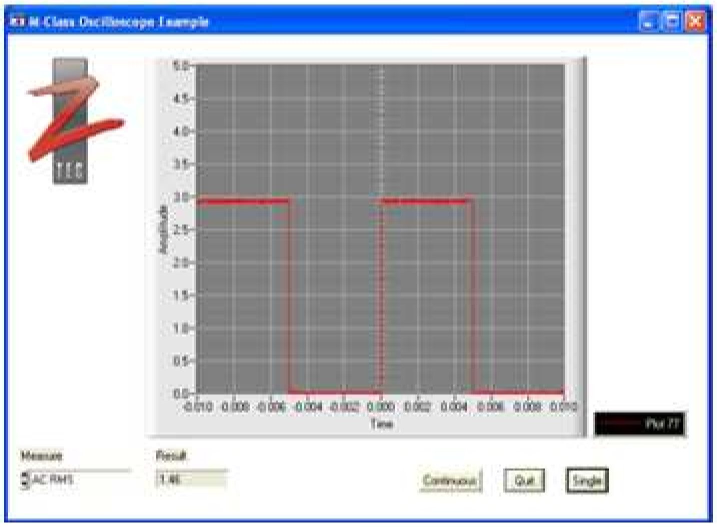
For this subway electronics test system, DiagnoSYS used PXI which is mounted in a 19-inch rack. Within this system, the ZTEC ZT450 PXI is used mainly to test the functionality and accuracy of the train's speed sensing equipment. This equipment consists of a microwave Doppler box that generates a 26 GHz signal which is focused onto the train track. The Doppler box then receives the reflected signal and is able to sense the frequency shift between the generated and received signal. Using the Doppler principle--where the perceived frequency (reflected) wave changes in proportion to the speed of the object that is generating the wave—the Doppler box generates a differential sine wave with a frequency and amplitude in proportion to the speed of the object at around 2 KHz. This signal is then sent to the train's speed sensor or digital speedometer which displays the speed of the subway car.

**Figure 1: DiagnoSYS Test System**

To test the functionality of this speed sensing equipment, the PXI digitizer is placed between the Doppler box and the speedometer and acquires the 2 KHz differential signal. The positive signal is routed into one channel on the digitizer and the negative signal is routed into the second channel. DiagnoSYS uses an onboard calculation channel to subtract the two channels to acquire the differential signal. Then they use the ZT450 PXI onboard measurements to gather the peak-to-peak voltage, frequency, and RMS of the signal. In software, they can compare these measurements to the expected values to verify that the Doppler box and speed sensor are working correctly.



```
ZT_HANDLE instr_handle;
int main (int argc, char *argv[])
{

float smpl_rt, off_ref, off_tm;
int smpl_pts;




char measure[100];


    if (InitCVIRTE (0, argv, 0) == 0)
        return -1;  /* out of memory */
    if ((panelHandle = LoadPanel (0, "C-Class Basic_Example.uir", PANEL)) < 0
        return -1;
    error = ztscopeC_init (RESOURCE_NAME, ZTSCOPEC_DO_ID_QUERY, ZTSCOPEC_RESET
    error = ztscopeC_channel_enable (instr_handle, ZTSCOPEC_INPUT_CH1, ZTSCOPE
    error = ztscopeC_channel_enable (instr_handle, ZTSCOPEC_INPUT_CH2, ZTSCOPE
    error = ztscopeC_horizontal (instr_handle, NUMPOINTS , 10e6, 0, 0);
    error = ztscopeC_vertical (instr_handle, ZTSCOPEC_INPUT_CH1, 50.0, 0.0, ZT
    error = ztscopeC_acquisition (instr_handle, ZTSCOPEC_ACQ_NORM, 8, ZTSCOPEC
    error = ztscopeC_trigger (instr_handle, ZTSCOPEC_INPUT_CH1, 1.0, ZTSCOPEC_

    DisplayPanel (panelHandle);
    RunUserInterface ();
    ztscopeC_close(instr_handle);
    DiscardPanel (panelHandle);
return;

}


int CVICALLBACK measurement (int panel, int control, int event,
        void *callbackData, int eventData1, int eventData2)
{
    switch (event)
    {
        case EVENT_COMMIT:

    SetCtrlVal (PANEL,PANEL_NUMERIC, result);
```



Measure: AC RMS

Result: 1.46

Figure 2: ZTEC Oscilloscope Drivers