ISSN:2231-0711
www.ijcset.net

Signal Generation Using TMS320C6713 Processor

Dolly Reney [1] *and* Dr.Neeta Tripathi [2]
[1] *Assist. Prof.E &TC Deptt.C.CE&T, Bhilai, INDIA*
[2] *Principal SSITM Bhilai, INDIA*

Abstract: **- This project focuses on developing a software user interface using Matlab tools that will allow a user to easily execute C, C++ and assembly language code on a Texas Instrument DSP board (TMS320C6713). A signal processing description to be executed on this DSP board will first be implemented using either a Matlab M-file or Simulink block diagrams. The block diagram in Simulink is first converted to C code using the Real-Time Workshop feature in Simulink. The Code Composer Studio software package converts the C code into an executable file, which is downloaded onto the board and displayed on the oscilloscope**.

Keywords: **- MATLAB, Simulink, TMS320C6713, DSP, RTW**

I. INTRODUCTION

A different approach of understanding Digital Signal Processing [5] which uses design of DSP Simulink blocks concepts is presented in this paper. This different approach focuses on DSP problems and concepts using TIC6000 and DSP Simulink block sets, in fact avoiding the use of algorithm such as assembly and C. One of the major attributes of the proposed applications is its flexibility to conduct hands-on experiments with real signals using Texas instruments TMS320C6713 evaluation boards in Digital Signal Processing educational lab. The applications developed are based on Simulink models which are built up with the help of TIC6000 and DSP Simulink block set. Once any of the models is complete and built, it automatically opens Code Composer Studio and compiles the Simulink block sets model. When no compilation error is detected, TMS320C6713 automatically interfaces with the real world and is ready for testing.

II. FUNCTIONAL DESCRIPTION

A. System Block Diagram

The software interface to be developed for the code integration will operate according to the block diagram shown below in Figure 1.

A.1.Signal Processing Description

This is a given problem to be implemented on the D.S.P board. It can involve implementing a control feedback system on the D.S.P board or analyzing a complex data acquisition scheme on a D.S.P board. Any problem whose output is in the form of signals or data that can adequately be processed in real time on the TMS3206713 is our Signal Processing Description.

A.2. Host PC with MATLAB Development Tools

The signal processing description is implemented by first writing code using Matlab development tools. If the signal processing description is in C++, C or assembly language code, it has to be converted to a MEX file, which is interfaced with Matlab. The Matlab code generated is saved into an input file or a workspace to be used in Simulink.

A.3. Simulink

Simulink, an extension of Matlab will be used also to solve our Signal Processing Description by first obtaining from Matlab an input file and displaying our graphical results numerically. A number of design blocks which may be unavailable in Simulink need to be developed in Matlab by writing C code. Only a constructed block diagram representation of the Signal Processing Description in Simulink is sent to the Code Composer Studio (CCS).

A.4.Code Composer Studio (CCS)

The CCS integrated development software converts the Simulink block diagram into C and assembly language, which downloaded as an output file onto the TMS3206713. The running process can be accessed only from the CCS debugging tools or across a link for CCS or Real-time data Exchange. Otherwise the running process is not accessed.

A.5.D.S.P Board (TMS3206713)

The TMS3206713 replaces the work of Simulink. The TMS3206713 outputs real time processed data or signals to the oscilloscope. The output on the scope represents our signal processing result.

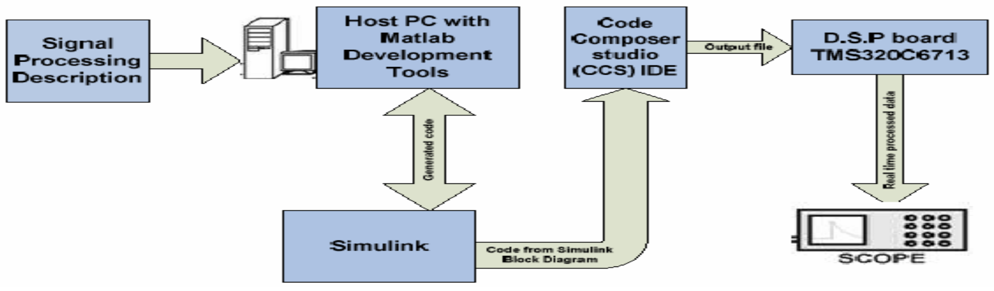

Signal Processing Description
Host PC with Matlab Development Tools
Code Composer studio (CCS) IDE
Output file
D.S.P board TMS320C6713
Generated code
Real time processed data
Simulink
Code from Simulink Block Diagram
SCOPE

Figure 1 is a High-level system block diagram

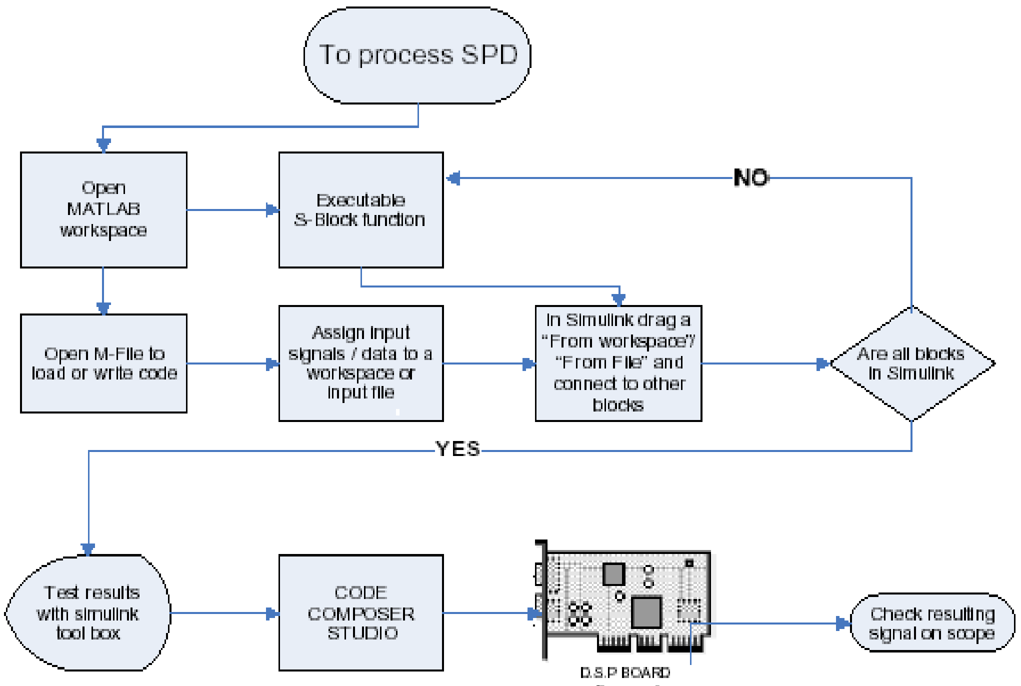

To process SPD
Open MATLAB workspace
Executable S-Block function
NO
Open M-File to load or write code
Assign input signals / data to a workspace or input file
In Simulink drag a "From workspace"/ "From File" and connect to other blocks
Are all blocks in Simulink
YES
Test results with simulink tool box
CODE COMPOSER STUDIO
D.S.P BOARD
Check resulting signal on scope

Figure 2 is a system flow chart

III. SYSTEM FLOW CHART

A system flow chart explaining the interconnection between Matlab Simulink and the DSP board is shown in above Figure 2.

IV.RESULT AND SIMULATION.

Creating models using input workspaces/files

1. A simple sine wave was created and stored in a workspace variable called simin as in an M-file:

PROGRAM:-

npts=2000; % these are the total number of pts per cycle in our simulation
n=0:1:npts-1;% number of pts n are in the increments of n=0,1,2,3...1999
fss=8000; % this is the sampling frequency for the input
delta=1/fss; % this will be used in simulation purposes
t=n*delt; % Total time t=1999/8000=0.249 figure(1),
Vin=sin(2*pi*2000*t); % this is the analog signal
plot(t,Vin),grid on % this plots Vin for the specified t range
xlabel('time(seconds)'); % add axis labels and plot title
ylabel('magnitude of sine wave');
title('A simple sine wave of freq=2000Hz');
whos; %display the contents of all variables used in matlab workspace
simin=[t' Vin'] % the transpose of t and Vin is saved in thevariable simin
save myinput.mat % the values of t and Vin saved in the variable in an input file**.**

2. Having done this a new model was created to transfer data from Matlab to Simulink as:
In the Simulink browser select and drag to the Simulink model,
Simulink>>Sources>>From Workspace block>>
Click on the From Workspace block and specify under data **simin** and 1/8000 as sampling time

3. A simple low-pass filter whose cut-off frequency was **714 Hz** was designed with the following Transfer Function:
H(s) = 714/(s+714)
In the Simulink browser select and drag to the Simulink model,
Sources>>Continuous>> Transfer Fuc block
Click on the **Transfer Fuc block** and specify the filter in the form of a matrix:
Num= [714]
Den= [1 714]
And the absolute tolerance in this same window is specified as **auto.**

4. Having completed the filter specifications a workspace variable will be selected to store the output from the filter.
Simulink>>Sources>>To Workspace block>>
Click on the To Workspace block and specify under data **simout** and 1/8000 as sampling time.

5. This model is simulated for **0.25 sec**, which is specified in the simulation parameter stop time window.
The model can now be simulated and analyzed using the scope block found at:
Simulink>>Sinks>>Scope

6. The c6713DSK was dragged to this model and the Realtime Workshop was used start the implementation of our model on the DSP board.
The build up process was interrupted by an error message that only asked for discrete (no continuous-state) blocks to be in the model. The filter whose transfer function was specified as an H(s) had to be converted to an H(z) using bilinear Transformation.
H(z)=H(s)|**s=2*fss*(z-1/z+1),**
Hence H(z) = (714z+714)

(16714z-15286)
This discrete transfer function replaces the previous transfer function by choosing:
>>Simulink>>Discrete>>Transfer Func Block
and specifying these parameters:
Num= [714 714]
Den= [16714 -15286]
Sampling time can be specified as -1 to inherit the sampling frequency used in the M-file code or that specified in simin block.
Having completed this, a Simulink model was created as shown below:

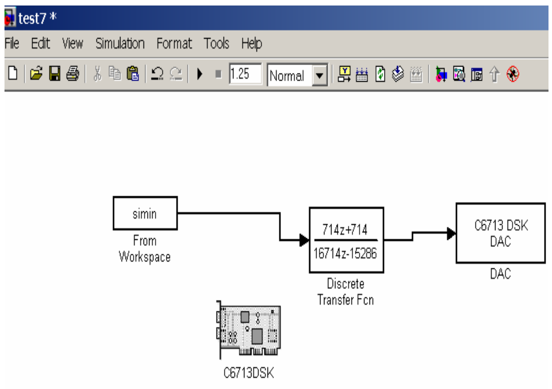


Fig. Model for generating Sine wave

This model is compiled and built using the Real-Time Workshop and the output of the scope is observed for the time duration of the simulation as:

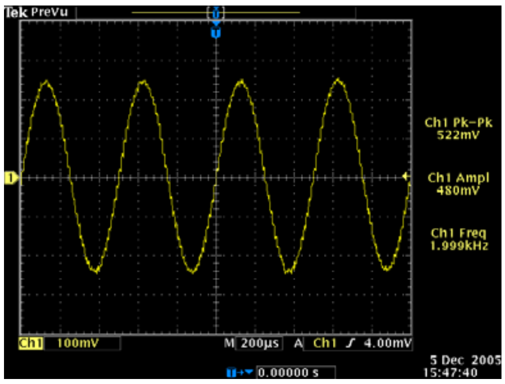


Fig.Output as in Oscilloscoe

A much better approach would have been to design a model that runs for an infinite amount of time using the block in the Simulink Browser window as follows:
>> Fixed-Point Blockset>>Sources>>Repeating sequence Stair
Upon clicking on the Repeating sequence stair block, the vector or output variables must be specified as [Vin'] and sampling time must be -1 to inherit the sampling frequency of fss=1/8000. Specifying the output variable as Vin' will store the values of our sine wave in a loop in this block, which can be effectively simulated for an infinite amount of time on the DSP board.

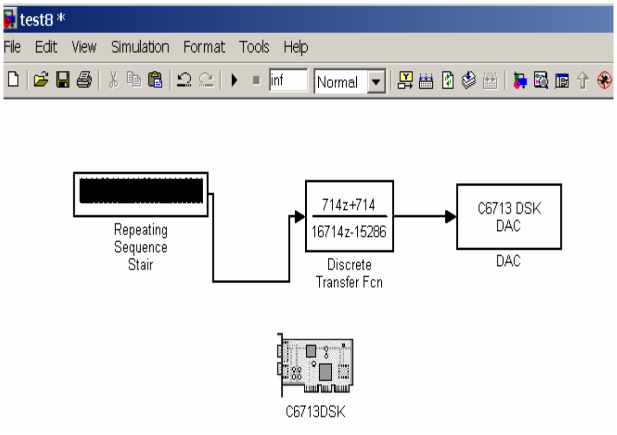
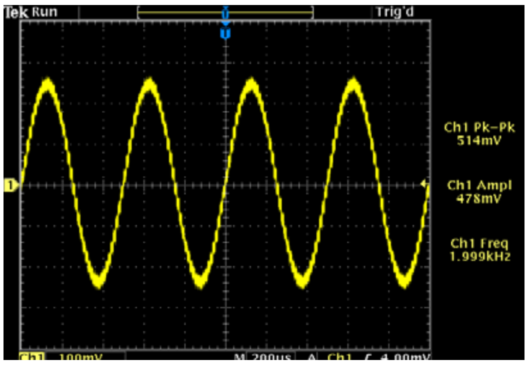


Fig. Model to generate output for infinite amount of time.

This model is compiled and built using the Real-Time Workshop and the output of the scope is observed for an infinite amount of time shown below in the scope plot below



V. Conclusion

There are many applications for which the Digital Signal Processor becomes an ideal choice as they provide the best possible combination of performance, power and cost. By using digital signal processing flexible signal generator that are often used in measuring and testing of communication systems is implemented. The output frequency range can be increased by using an external high-speed digital-toanalogue converter. The DSP processor TMS320C6713DSK with Code Composer Studio and C programming language has been used to generate the desired waveforms. Two different types of waveforms can be generated simultaneously using the stereo codec of TMS320C6713DSK. Because of the high performance of DSP processor and the efficiency of the C language, signal generation has become user-friendly and more programmable.

REFERENCE

[1] TMS320C6713 DSK Technical Reference, Spectrum Digital, Inc., May 2003.

[2] Code Composer Studio Help, Texas Instrument, 2003

[3] Rajean Arseneau, Michelle E. Sutherland, and J.J. Zelle,'A Test System for Calibrating Flicker meters', IEEE Transactions Instrumentation and Measurement, Vol. 51, No. 4, August 2002.

[4] Sia Lih Huoy, S.S. Jamuar, Roslina Mohd. Sidek, and Mohd.Hamiruce Marhaban, 'Digital Signal Processing Based Waveform Generator for Flicker meter Calibration Test System'4th Student Conference on Research and Development (SCOReD 2006), Shah Alam, Selangor, MALAYSIA, 27-28 June, 2006, IEEE 2006

[5] TMS320C6000 Optimizing Compiler – User's Guide, Texas Instruments Literature Number SPRU187L,May 2004.

[6] TMS320C6713, TMS320C6713B Floating-Point Digital Signal Processors, Texas Instruments Manual SPRS186I, December 2001, Revised May 2004.

[7] http://www.electronicsprojectdesign.com/SignalGenerator

[8] H. Sia, S.S. Jamuar, Roslina Mohd Sidek and Mohd Hamiruce Marhaban 'Digital-signal-processor-based waveform generator' IOP PUBLISHING, Meas. Sci.Technol. 18 (2007) N35–N40

[9] E.A. Lee, Programmable DSP Architectures: Part I, *IEEE ASSP Mag*., October 1988.

[10] TMS320C621x/C671x DSP Two-Level Internal Memory Reference Guide, Texas Instruments Literature Number SPRU609A, November 2003.

[11] D. Talla, L.K. John, V. Lapinskii and B.L. Evans, Evaluating Signal Processing and Multimedia Applications on SIMD, VLIW and Superscalar Architectures, Proceedings of the International Conference on Computer Design, September 2000.