Tutorial DS1104

Emmanuel Guillermo Pérez

August 2022

To-do List

- ☐ Abstract
- ☐ Introduction
- ☐ Controller board
  - ☐ Types of controller boards
- ☑ Rapid control prototyping
- ☑ Hardware in the loop simulation
- ☐ dSPACE Controller Board
- ☑ Hardware-in-the-loop demonstration for driving a servomotor PWM dSPACE
- ☐ Ball and beam Simulink design - implementation
- ☐ Ball and beam hardware implementation
- ☐ KiCAD
- ☐ References
- ☐ Appendix´s

Abstract

It´s a short overview of what the paper entails. No more than 6 lines.

1 Introduction

2 Controller Board

A piece of hardware that is responsible for capturing, processing and sending signals or information to the system to be controlled. It can be made up of elements such as:

- Microcontroller/s to process the data.
- I/O modules
- Data bus modules, essentials for internal and external communication of the board.
- A/D signal processing modules
- Timers

This is where the firmware is housed, a specific class of computer software that provides the low-level control for a device's specific hardware.

3 Rapid control prototyping

Rapid control prototyping (**RCP**) is one of the most efficient ways to speed up the study and development of a new product. **RCP** greatly simplifies moving from the design to the implementation stage of a control system by quickly verifying how it will respond to real-world dynamics.[2]
A great benefit of **RCP** systems is that they eliminate the tedious and error-prone process of low-level programming because they have automatic code generation capability, which gives engineers the ability to focus on the design, implementation and evaluation of the control system.

Several companies offer software and hardware solutions that allow the design of control systems using a block diagram programming paradigm. Among them, MATLAB/Simulink is probably the best known and most widely used simulation software. MATLAB is a high-level computer language for algorithm development, visualization and data analysis, while Simulink is an interactive tool for modeling, simulating and analyzing dynamic systems.
Simulink's companion product, Real-Time Workshop (**RTW**), provides automatic ANSI-C or ADA code generation from the Simulink block diagram. **RTW** is not hardware-specific, so the generated code can be deployed on a wide range of personal computers, digital signal processors, or even microcontrollers.

To prevent the wasting of the development team time with hardware limitations, conventional **RCP** systems must have three key elements:

1. A powerful floating-point processor, several times faster than the target processor.
2. Different types of flexible I/O.
3. A big memory.

Control boards, such as the DS1104, are appropriate for motion control and are fully programmable from the Simulink environment. These large-scale **RCP** systems are very

powerful and suitable for applications where functionality takes priority over price, such as in research.

In the educational process, for example, the least efficient, cost-effective and portable **RCP** solutions are used. **RCP** systems for educational purposes should also be as simple to use as possible. If so, students can focus on designing and verifying control systems rather than learning how to operate a **RCP** system. Such **RCP** systems are difficult to find on the market, so institutions sometimes choose to develop custom internal solutions.

4 Hardware in the loop simulation

4.1 What is a simulation?

It is an experiment performed on a model. A model is a mathematical representation of a physical system without discriminating whether its nature is mechanical, electrical or a combination of both, which allows to express in the form of numbers, constants and variables, any physical magnitude.
The main advantage is that the behavior of a system can be studied without the need to do it physically. If we take as an example an AC electrical system whose voltage is higher than 24 V, any direct contact would be dangerous. However, the result of the simulation it´s completely dependent on how well the model represents the real system.
Simulations can been performed in different domains depending on the modeling philosophy used to model the system. It can be time-domain or frequency domain.[1]

4.2 Real Time simulation

A real time simulation it´s a time-domain simulation in which the independent variable, that it´s time, grows at the same pace as the actual time does. So, if we want some device or system to properly interact with the simulation, we should make sure that the simulation runs in real time.
There are two types of RT-HIL simulation.

- **Close-loop**: in which there is a two-way flow data between the RT-simulator and the device. This type of RT-HIL simulation is suited to control and protection applications.
- **Open-loop**: in which there is a one-way flow of data, typically from the RT-simulator to the device. This type of RT-HIL simulation is suited for monitoring applications.

4.3 dSPACE HIL simulation

When your simulated controller is able to control your real plant, you typically produce the actual controller. For the final tests you usually connect the real controller to a model of the plant, which, of course, has to be simulated in real time. This way you can ensure that the controller does not contain any errors that could damage the real plant.

This technique is called *hardware-in-the-loop simulation* (**HIL**). For both **RCP** and **HIL** the real-time simulation is rather important. The computing power required by real-time simulation highly depends on the characteristics of the simulated model: If it contains very demanding calculations you have to provide a lot of computing power because the timing cannot be satisfied otherwise. **dSPACE** systems fulfill this demand for computing power.

5 dSPACE Controller Board

5.1 General description

The dSPACE system is high performance digital control system based on the MPC8240 processor and the TMS320F240 DSP processor, a breakout panel and the software tools. It is directly interfaced with MATLAB/SIMULINK running on a PC. a SIMULINK block diagram is converted to real time C and plotting variables in real time in the DSP.

5.1.1 Board Architecture

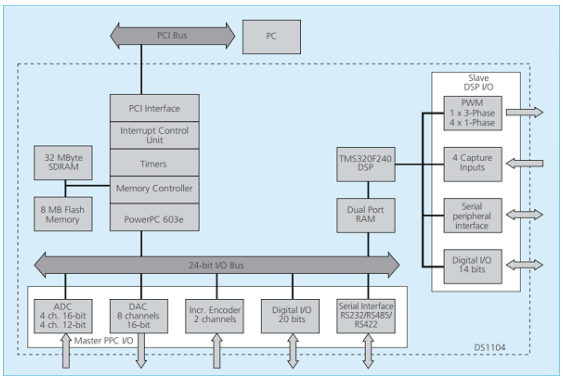

Figure 1: DS1104 architecture

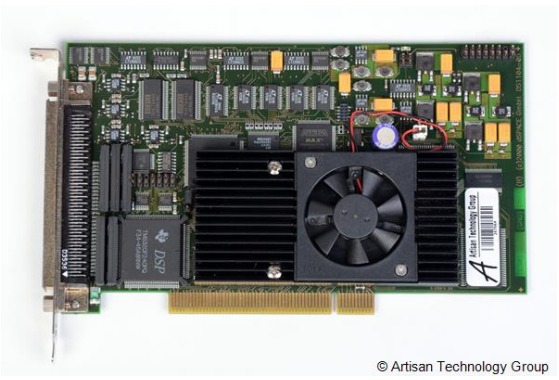

Figure 2: DS1104 Control Board

5.1.2 Connection to External Devices

There are three different ways to connect external devices to the DS1104. To access the I/O units of the master PPC and the slave DSP, connect external devices

- to the 100-pin I/O connector P1 of the DS1104, or
- to the adapter cable with two 50-pin Sub-D connectors P1A and P1B, that are included in the DS1104 hardware package, or
- to the optional connector panel CP1104 or the optional combined connector/LED panel CLP1104, which provides an additional array of LEDs indicating the states of the digital signals.

5.1.3 Memory Features

The DS1104 is equipped with two memory sections:

- Global memory

  32-MByte synchronous DRAM (SDRAM) for applications and data.

  Fully cached (L1 cache).

- Flash memory

  8 MByte, divided into 4 blocks of 2 MByte each.

  6.5 MByte can be used for a user-specific application.

  1.5 MByte are reserved for the boot firmware.

  8-bit read / write access by master PPC.

At least 100,000 erase cycles possible.

5.1.4 Timer Features

The DS1104 board is equipped with 6 timer devices. The timers are driven by the bus clock, whose frequency is referred to as BCLK.
Using ControlDesk Next Generation, you can get the current BCLK value via the Properties controlbar.
The timers have the following characteristics:

- Time Base Counter

  Free-running 64-bit up counter driven by BCLK/4

  Used for measurement of relative and absolute times

  Used for time-stamping

- Timers 0 . . . 3

  32-bit down counters driven by BCLK/8

  Used as trigger source for periodic tasks

  When the counter reaches 0, the timer generates an interrupt and the counter is reloaded with the value of the period register.

- Decrementer

  32-bit down counter driven by BCLK/4

  Used as trigger source for periodic tasks

  When the counter reaches 0 the timer generates an interrupt. A new counter value is set by software in the timer interrupt service routine.

5.1.5 Timer names

The names of the timers listed above are the names of the hardware timer devices used by RTLib. They differ from the names used by RTI Timer Interrupt block:

Timer Device Name Used by RTLib	Timer Interrupt Name Used by RTI
Timer 0	Timer A interrupt
Timer 1	Timer B interrupt
Timer 2	-(not supported by RTI)
Timer 3	-(not supported by RTI)
Decrementer	Timer C interrupt

Table 1: dSPACE Timers

5.1.6 Timer interrupts for periodic events

Timers 0 . . . 3 and the Decrementer provide timer interrupts that you can use to trigger periodic events in a real-time application. These 32-bit down counters generate an interrupt whenever they reach 0. Then the timer is automatically reloaded.

5.1.7 Host Interface

The DS1104 provides a PCI interface requiring a single 5 V PCI slot. The interface has the following characteristics:

- Access from/to the host PC via 33 MHz-PCI interface.
  The interface serves the board setup, program downloads and runtime data transfers from/to the host PC.
- Interrupt line
  The host interface provides a bidirectional interrupt line: Via this line, the host PC can send interrupt requests to the master PPC and vice versa. Both the host PC and the master PPC can monitor the state of the interrupt line to detect when the corresponding interrupt service is finished.

5.2 Features Provided by the Master PPC

The DS1104's main processing unit, MPC8240, consists of:

- A PowerPC 603e microprocessor (master PPC) on which the control models will be implemented.

  Running at 250 MHz (CPU clock)

  Containing a 16-KByte L1 data cache

  Containing a 16-KByte L1 instruction cache
- An interrupt controller
- A synchronous DRAM controller
- Several timers
- A PCI interface (5 V, 32 bit, 33 MHz)

The master PPC controls the following I/O features of the DS1104:

- ADC Unit
- DAC Unit
- Bit I/O Unit
- Incremental Encoder Interface
- Serial Interface

5.2.1 ADC Unit

The master PPC on the DS1104 controls an ADC unit featuring two different types of A/D converters:

- 1 A/D converter (ADC1) multiplexed to four channels (signals ADCH1 . . . ADCH4).The input signals of the converter are selected by a 4:1 input multiplexer. The A/D converters have the following characteristics:

  16-bit resolution

  ±10 V input voltage range

  ± 5 mV offset error

± 0.25 % gain error

> 80 dB (at 10 kHz) signal-to-noise ratio (SNR)

- 4 parallel A/D converters (ADC2 . . . ADC5) with one channel each (signals ADCH5 . . . ADCH8). The A/D converters have the following characteristics:

  12-bit resolution

  ±10 V input voltage range

  ± 5 mV offset error

  ± 0.5 % gain error

  > 70 dB signal-to-noise ratio (SNR)

Read modes

The A/D converters can be used in polling and in non-polling mode. In polling mode, the conversion values can be read if the end-of conversion flag in the ADC control register is set to 1. In non-polling mode, the conversion values are read immediately without waiting on the completion of the conversion. The non-polling functions are *ds1104_adc_read_ch_immediately* and *ds1104_adc_read_conv_immediately*.

Signal mapping

Signal	Channel/Bit Numbers of Related RTI Blocks/RTLib Functions				I/O Pin on . . .		
	Related RTI Block(s)	Ch/Bit (RTI)	Related RTLib Functions	Ch/Bit (RTILib)	DS1104	Sub-D Conn.	CP/CLP
ADC Unit							
• Imput voltage range: ± 10 V • ADCH1 ... ADCH4: input for A/D converter with 16-bit resolution • ADCH5 ... ADCH8: input for A/D converter with 12-bit resolution							
ADCH1	DS1104MUX_ADC	Ch 1	See ADC Unit	Ch 1	P1 100	P1A 50	CP1
ADCH2		Ch 2		Ch 2	P1 99	P1B 50	CP2
ADCH3		Ch 3		Ch 3	P1 96	P1A 33	CP3
ADCH4		Ch 4		Ch 4	P1 95	P1B 33	CP4
ADCH5	DS1104ADC_Cx	Ch 5	See ADC Unit	Ch 5	P1 92	P1A 16	CP5
ADCH6		Ch 6		Ch 6	P1 91	P1B 16	CP6
ADCH7		Ch 7		Ch 7	P1 88	P1A 48	CP7
ADCH8		Ch 8		Ch 8	P1 87	P1B 48	CP8

Table 2: ADC Signal mapping

Input circuit

The following illustration is a simplified diagram of the input circuitry of the ADCs.

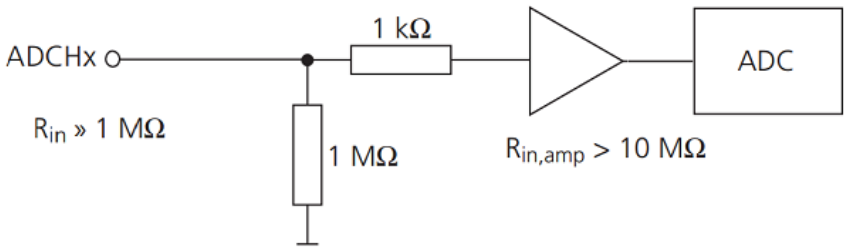

Figure 3: ADCs Input Circuit

RTI/RTLib support

You can access the master PPC's ADC unit via RTI1104 and RTLib1104.
For details, see:

- ADC Unit in the DS1104 RTI Reference
- ADC Unit in the DS1104 RTLib Reference

DS1104MUX_ADC

The purpose of this block is to read up to four channels of the A/D Converter specifying the channels to be multiplexed. The width of the block output vector (comprising the selected channels assigned to one converter) matches the number of the selected channels.

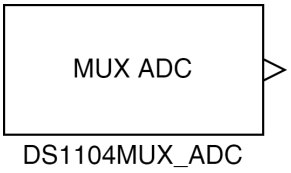

Figure 4: MUX ADC Block

DS1104ADC_Cx

The purpose of this block is to read from a single channel of one of 4 parallel A/D converter channels. Lets you select a single channel within the range 5 ... 8 and it´s scales between the analog input voltage and the output of the block:

Input Voltage Range	Simulink Output
–10 V ... +10 V	–1 ... +1 (double)

Table 3: ADC I/O characteristics

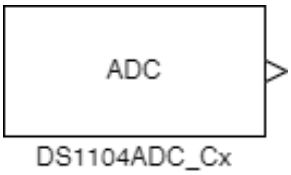

Figure 5: ADC_Cx Block

5.2.2 DAC Unit

The master PPC on the DS1104 controls a D/A converter. It has the following characteristics:

- 8 parallel DAC channels (signals DACH1 ... DACH8)
- 16-bit resolution
- ± 10 V output voltage range
- ± 1 mV offset error, 10 μV/K offset drift

- ± 0.1% gain error, 25 ppm/K gain drift
- > 80 dB (at 10 kHz) signal-to-noise ratio (SNR)
- Transparent and latched mode

Transparent and latched mode

The DAC unit of the master PPC can be driven in two operating modes:

- In the *transparent mode*, the converted value is output immediately.
- In the *latched mode*, the converted value is output after a strobe signal. This allows you to write output values to more than one channel, and output the values simultaneously.

Synchronization with ST1PWM signal

Updating DAC outputs can be synchronized with PWM signal generation or an external trigger source.

I/O - signal mapping

The following table shows the mapping between the RTI block and RTLib functions and the corresponding pins used by the DAC unit.

Signal	Channel/Bit Numbers of Related RTI Blocks/RTLib Functions				I/O Pin on ...		
	Related RTI Block(s)	Ch/Bit (RTI)	Related RTLib Functions	Ch/Bit (RTILib)	DS1104	Sub-D Conn.	CP/CLP
DAC Unit							
• Output voltage range: ± 10 V • Output current range: ± 5 mA • DACH1 ... DACH8: D/A converter output with 16-bit resolution							
DACH1	DS1104DAC_Cx	Ch 1	See DAC Unit	Ch 1	P1 84	P1A 31	CP9
DACH2		Ch 2		Ch 2	P1 83	P1B 31	CP10
DACH3		Ch 3		Ch 3	P1 80	P1A 14	CP11
DACH4		Ch 4		Ch 4	P1 79	P1B 14	CP12
DACH5		Ch 5		Ch 5	P1 76	P1A 46	CP13
DACH6		Ch 6		Ch 6	P1 75	P1B 46	CP14
DACH7		Ch 7		Ch 7	P1 72	P1A 29	CP15
DACH8		Ch 8		Ch 8	P1 71	P1B 29	CP16

Table 4: DAC I/O - Signal mapping

Output circuit

The following illustration is a simplified diagram of the output circuitry of the DACs.

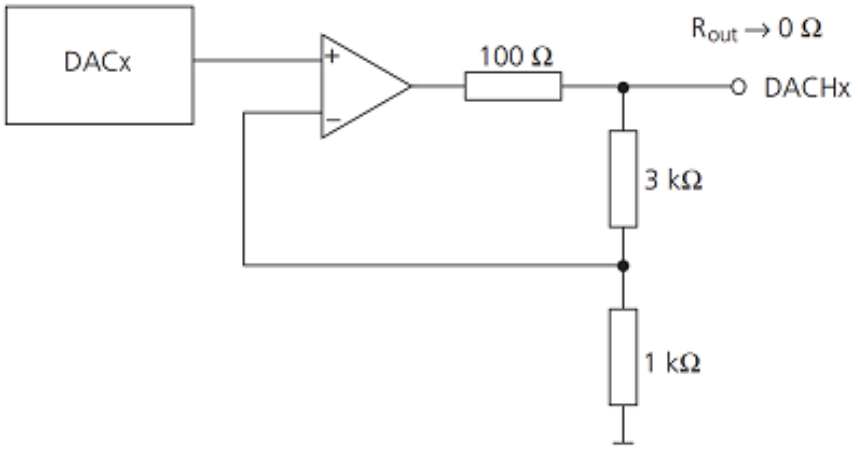

Figure 6: DACs Output Circuit

Electrical characteristics

The analog outputs are single-ended bipolar outputs with the following characteristics.

Parameter	Value	
	Min.	Max.
Output voltage	-10 V	+10 V
Output current	-5 mA	+5mA
Output resistance	$\rightarrow$ 0 Ω	
Power-up default	0 V	
SNR (signal-to-noise ratio)	>80 dB	

Table 5: DAC Electrical output characteristics

RTI/RTLib support

You can access the master PPC's DAC unit via RTI1104 and RTLib1104 to write to one of the 8 parallel D/A converter channels.

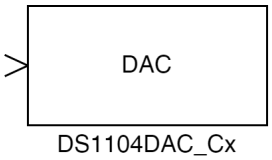

Figure 7: DAC_Cx Block

I/O characteristics

- Scaling between the analog output voltage and the input of the block:

Simulink Input	**Output Voltage Range**
–1 ... +1 (double)	–10 V ... +10 V

Table 6: DAC I/O characteristics

- The block provides its outputs in:

  Transparent mode, that is the channel is converted and output immediately.

  Latched mode, that is the channel is converted after synchronous triggering.

Initialization

During the model initialization phase, an initial output voltage value is written to each D/A channel. This is especially useful if a channel is used within a triggered or enabled subsystem that is not executed right from the start of the simulation.

With the initialization value, the channel has a defined output during this simulation phase.

In other words, lets you specify the initial output voltage at the start of the simulation. The value must remain in the output voltage range ±10 V.

Termination

When the simulation terminates, the channel holds the last output value by default. You can specify a user-defined output value on termination, and use these settings to drive your external hardware into a safe final condition.

The specified termination values of I/O channels are set when the simulation executes its termination function by setting the simState variable to STOP. If the real-time process is stopped by using ControlDesk's Stop RTP command, the processor resets immediately without executing termination functions. The current values of the I/O channels are kept and the specified termination values are not set.

Output on termination lets you either keep the current output voltage when the simulation terminates, or set the output to a specified value. The value must remain in the output voltage range ±10 V

5.2.3 Bit I/O Unit

The master PPC on the DS1104 controls a bit I/O unit with the following characteristics:

- 20-bit digital I/O
- Direction selectable for each channel individually
- ±5 mA maximum output current
- TTL voltage range for input and output
- You can also use the bit I/O unit provided by the slave DSP, which contains 14-bit digital I/O.

I/O characteristics

Relationship between the digital input and the output of the DS1104BIT_IN_Cx block:

Digital Input (TTL)	Simulink Output	
	Without Data Typing	With Data Typing
High	1 (double)	1 (boolean)
Low	0 (double)	0 (boolean)

Table 7: I/O Input characteristics

Relation between the digital output and the input of the DS1104BIT_OUT_Cx block:

Simulink Output		Digital Output (TTL)
Without Data Typing	With Data Typing	
>0 (double)	1 (boolean)	High
≤0 (double)	0 (boolean)	Low

Table 8: I/O Output characteristics

Description

During the model initialization phase, an initial digital output value is written to each channel. This is especially useful if a channel is used within a triggered or enabled subsystem that is not executed right from the start of the simulation. With the initialization value, all channels have defined outputs during this simulation phase.

When the simulation terminates, all channels hold their last digital output values by default. You can specify a user-defined output value on termination, and use these settings to drive your external hardware into a safe final condition.

The specified termination values of I/O channels are set when the simulation executes its termination function by setting the simState variable to STOP. If you stop the real-time application by using ControlDesk Next Generation's Stop RTP (ControlDesk Next Generation Reference) command, the processor resets immediately without executing termination functions. The current values of the I/O channels are kept and the specified termination values are not set.

RTI block support

The master PPC library contains several blocks for programming the digital I/O unit.

To read from the digital I/O port

- Use DS1104BIT_IN_Cx to read certain bits of the digital I/O port. Lets you select a channel within the range 0 ... 19 where as the channels 16 ... 19 are multiplexed with 4 external interrupts.

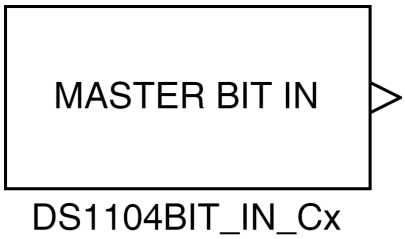

Figure 8: BIT_IN_Cx Block

To write to the digital I/O port

- Use DS1104BIT_OUT_Cx to write certain bits of the digital I/O port.

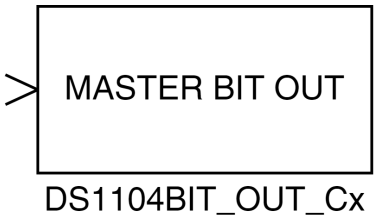

Figure 9: BIT_OUT_Cx Block

The following table shows the mapping between the RTI blocks and RTLib functions and the corresponding pins used by the Bit I/O unit.

Signal	Channel/Bit Numbers of Related RTI Blocks/RTLib Functions			I/O Pin on ...		
	Related RTI Blocks	Ch/Bit (RTI)	Ch/Bit (RTILib)	DS1104	CP/CLP	Sub-D Conn.
Bit I/O Unit						
• TTL voltage range • Output current range: ±5 mA						
IO0	DS1104BIT_IN_Cx	Bit 0	Bit 0	P1 68	CP17 20	P1A 12
IO1	DS1104BIT_OUT_Cx	Bit 1	Bit 1	P1 67	CP17 2	P1B 12
IO2		Bit 2	Bit 2	P1 66	CP17 21	P1A 28
IO3		Bit 3	Bit 3	P1 65	CP17 3	P1B 28
IO4		Bit 4	Bit 4	P1 64	CP17 23	P1A 44
IO5		Bit 5	Bit 5	P1 63	CP17 5	P1B 44
IO6		Bit 6	Bit 6	P1 62	CP17 24	P1A 11
IO7		Bit 7	Bit 7	P1 61	CP17 6	P1B 11
IO8		Bit 8	Bit 8	P1 60	CP17 26	P1A 27
IO9		Bit 9	Bit 9	P1 59	CP17 8	P1B 27
IO10		Bit 10	Bit 10	P1 58	CP17 27	P1A 43
IO11		Bit 11	Bit 11	P1 57	CP17 9	P1B 43
IO12		Bit 12	Bit 12	P1 56	CP17 29	P1A 10
IO13		Bit 13	Bit 13	P1 55	CP17 11	P1B 10
IO14		Bit 14	Bit 14	P1 54	CP17 30	P1A 26
IO15		Bit 15	Bit 15	P1 53	CP17 12	P1B 26
IO16		Bit 16	Bit 16	P1 52	CP17 32	P1A 42
IO17		Bit 17	Bit 17	P1 51	CP17 14	P1B 42
IO18		Bit 18	Bit 18	P1 50	CP17 33	P1A 9
IO19		Bit 19	Bit 19	P1 49	CP17 15	P1B 9

Table 9: Bit I/O Unit mapping

6 Features Provided by the Slave DSP

The DS1104's slave DSP subsystem consists of

- A Texas Instruments TMS320F240 DSP

  Running at 20 MHz

  4Kx16 bit dual-port memory (DPMEM) used for communication with the master PPC

6.1 Slave DSP Bit I/O Unit

The slave DSP on the DS1104 provides a bit I/O unit with the following characteristics:

- 14-bit digital I/O
- Direction selectable for each channel individually
- ±13 mA maximum output current
- TTL voltage range for input and output

I/O mapping

The following table shows the mapping between the RTI blocks and RTLib functions and the corresponding pins used by the slave DSP bit I/O unit.

Signal	Channel/Bit Numbers of Related RTI Blocks/RTLib Functions			I/O Pin on ...		
	Related RTI Block(s)	Ch/Bit (RTI)	Ch/Bit (RTILib)	DS1104	Sub-D Conn.	CP/CLP
Slave DSP Bit I/O Unit						
• TTL voltage range • Output current range: ±13 mA						
SPWM7 *	DS1104SL_DSP_BIT_IN_Cx	Bit 0	Group 2 bit 0	P1 31	P1B 6	CP18 10
SPWM8 *	DS1104SL_DSP_BIT_OUT_Cx	Bit 1	Group 2 bit 1	P1 29	P1B 22	CP18 29
SPWM9 *		Bit 2	Group 2 bit 2	P1 27	P1B 38	CP18 11
ST1PWM *		Bit 3	Group 2 bit 3	P1 25	P1B 21	CP18 23
ST2PWM *		Bit 4	Group 2 bit 4	P1 23	P1B 37	CP18 5
ST3PWM *		Bit 5	Group 2 bit 5	P1 21	P1A 20	CP18 24
SCAP1 *		Bit 6	Group 3 bit 4	P1 18	P1A 20	CP18 2
SCAP2 *		Bit 7	Group 3 bit 5	P1 16	P1A 36	CP18 21
SCAP3 *		Bit 8	Group 3 bit 6	P1 14	P1A 3	CP18 3
SCAP4 *		Bit 9	Group 3 bit 7	P1 12	P1A 19	CP18 22
SCLK *		Bit 10	Group 4 bit 0	P1 17	P1B 20	CP18 17
SSTE *		Bit 11	Group 4 bit 1	P1 15	P1B 36	CP18 35
SSIMO *		Bit 12	Group 4 bit 2	P1 13	P1B 3	CP18 16
SSOMI *		Bit 13	Group 4 bit 3	P1 11	P1B 19	CP18 34

Table 10: Slave DSP Bit I/O Unit

6.2 Slave DSP Timing I/O Unit

The slave DSP on the DS1104 provides a timing I/O unit that you can use to generate and measure pulse-width modulated (PWM) and square-wave signals.

PWM signal generation

The PWM signal generation has the following characteristics:

- Outputs for the generation of up to four 1-phase PWM signals with variable:

Duty cycles (T/Tp ratio)

PWM frequencies

Polarity

Symmetric or asymmetric PWM mode

- Non-inverted and inverted outputs for 3-phase PWM signal generation (PWM3) with variable:

  Duty cycles (T/Tp ratio)

  PWM frequencies

  Deadband

- Non-inverted and inverted outputs for the generation of 3-phase space vector PWM signals (PWMSV) with variable:

  Values T_1 and T_2 of the space vector

  Sector of the space vector

  PWM frequencies

  Deadband

Square-wave signal generation (D2F)

The square-wave signal generation (D2F) provides outputs for the generation of up to four square-wave signals with variable frequencies.

PWM signal measurement (PWM2D)

The PWM signal measurement (PWM2D) provides inputs for the measurement of the duty cycle and frequency of up to four PWM signals. For details, see Slave DSP PWM Signal Measurement (PWM2D) on page 60.

Square-wave signal measurement (F2D)

The square-wave signal measurement (F2D) provides inputs for the measurement of up to four square-wave signals.

I/O mapping

Signal	Channel/Bit Numbers of Related RTI Blocks/RTLib Functions			I/O Pin on ...		
	Related RTI Block(s)	Ch/Bit (RTI)	Ch/Bit (RTILib)	DS1104	Sub-D Conn.	CP/CLP
1-Phase PWM Signal Generation (PWM), 3-Phase PWM Signal Generation (PWM3), Space Vector PWM Signal Generation (PWMSV)						
TTL output voltage range Output current range: ±13 mA						
ST2PWM *	DS1104SL_DSP_PWM	Ch 1	Ch 1	P1 23	P1B 21	CP18 5
SPWM7 *	DS1104SL_DSP_PWM3	Ch 2	Ch 2	P1 31	P1B 6	CP18 10
SPWM8 *	DS1104SL_DSP_PWMSV	Ch 3	Ch 3	P1 29	P1B 22	CP18 29
SPWM9 *		Ch 4	Ch 4	P1 27	P1B 38	CP18 11
SPWM1 *		Phase 1	Phase 1	P1 32	P1A 6	CP18 7
SPWM3 *		Phase 2	Phase 2	P1 28	P1A 38	CP18 8
SPWM5 *		Phase 3	Phase 3	P1 24	P1A 21	CP18 9
SPWM2		Phase 1 (inverted)	Phase 1 (inverted)	P1 30	P1A 22	CP18 26
SPWM4		Phase 2 (inverted)	Phase 2 (inverted)	P1 26	P1A 5	CP18 27
SPWM6		Phase 3 (inverted)	Phase 3 (inverted)	P1 22	P1A 37	CP18 28

Table 11: Slave DSP PWM Signal

6.3 Basics of Slave DSP PWM Signal Generation

The slave DSP of the DS1104 provides outputs for PWM signal generation. Each PWM pulse is centered around the middle of the corresponding PWM period (symmetric PWM generation mode).

PWM signals

PWM signal generation is crucial to many motor and motion control applications. PWM signals are pulse trains with fixed frequency and magnitude and variable pulse width. There is one pulse of fixed magnitude in every PWM period.

However, the width of the pulses changes from period to period according to a modulating signal. When a PWM signal is applied to the gate of a power transistor, it causes the turn-on/turn-off intervals of the transistor to change from one PWM period to another, according to the same modulating signal. The frequency of a PWM signal is usually much higher than that of the modulating signal, or the fundamental frequency, so that the energy delivered to the motor and its load depends mainly on the modulating signal.

PWM period, duty cycle and resolution

For PWM signals, you can specify the PWM period TP (= T_high+ T_low) in the range 200 ns ... 819,2 ms. For PWM3 and PWMSV signals, the PWM period TP applies to each of the 3 phases. For 1-phase PWM signals, the PWM period TP applies to each of the four PWM output channels. If you perform 3-phase and 1-phase PWM signal generation at the same time, you can specify different PWM periods for the 3-phase and 1-phase PWM signals.

You can also specify the duty cycle. The following illustration shows how the duty cycle d (= T_high/ TP) is defined. The available duty cycle range is 0 ... 1 (0 ... 100 %).

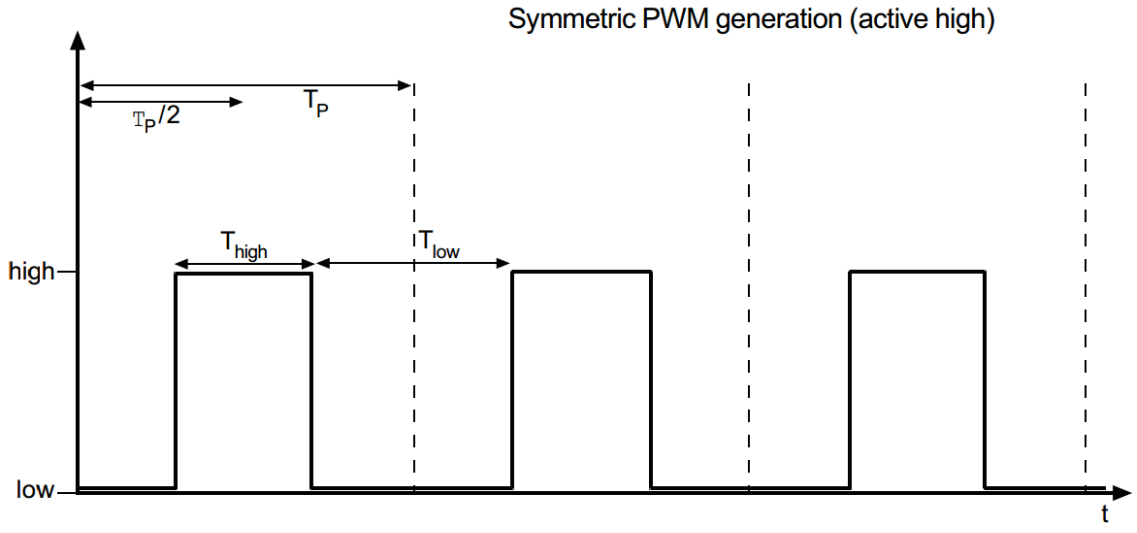

Figure 10: Symmetric PWM generation

Depending on the selected PWM period, the following resolutions are given. They apply to symmetric PWM signals.

Period T_p	Resolution
<6.4 ms	100 ns
<12.8 ms	200 ns
<25.6 ms	400 ns
<51.2 ms	800 ns
<102.4 ms	1.6 μs
<204.8 ms	3.2 μs
<409.6 ms	6.4 μs
<819.2 ms	12.8 μs

Table 12: Symmetric PWM resolution

Deadband

For the three PWM duty cycles of PWM3 and PWMSV, you can specify one deadband value. This is the time gap between the rising and falling edges of the non-inverted and inverted PWM signals. The deadband introduces a time delay that allows complete turning off of one power transistor before the turning on of the other power transistor.

The maximum deadband value is 100 μs. However, it should not be greater than $T_P/2$.

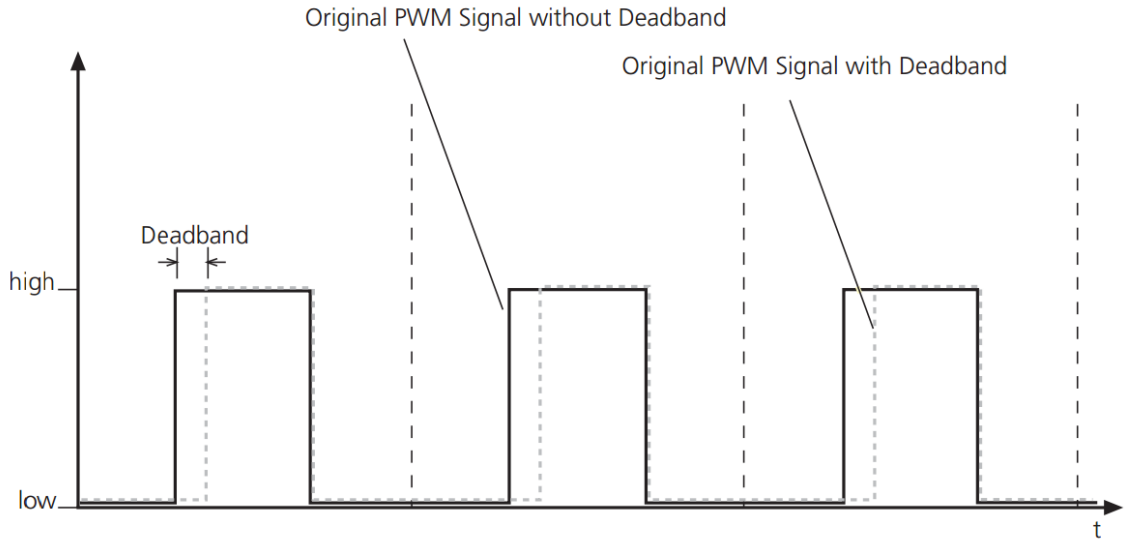

Figure 11: Original PWM Signal with Deadband

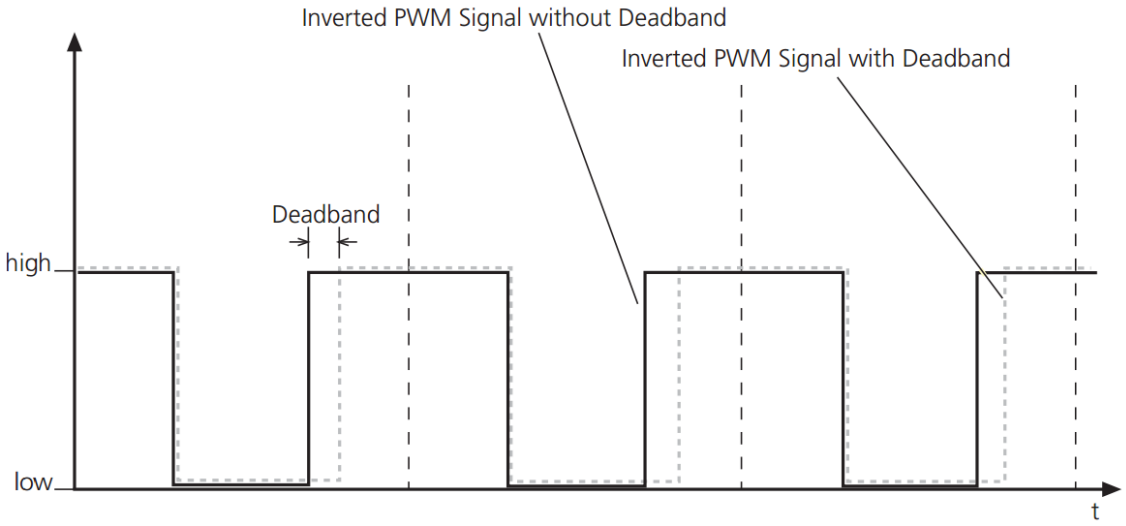

Figure 12: Inverted PWM Signal with Deadband

PWM outputs

For each of the PWM generation modes (1-phase, 3-phase and space vector), the PWM outputs can be specified. The running PWM generation can be suspended and the corresponding channels can be set to a specified TTL level (high or low). Only the output of the PWM signal is disabled. Signal calculation is still running and if you enable PWM generation, the currently calculated signal is output, and not the defined initialization or termination value. The PWM outputs can be specified for the two simulation phases (RTI):

- During the initialization phase, you can disable the PWM generation of selected channels (channel pairs for PWM3 and PWMSV) and set each output (pair) to a defined TTL level (high or low). No signal is generated during the initialization.
- During run time, you can stop PWM generation and set the outputs to a defined TTL level (high or low). At any time you can resume in generating the PWM signal. If the simulation terminates the outputs can be set to defined TTL levels.

If the PWM stop feature is disabled, the normal initialization and termination routines are executed. That means the specified duty cycles for initialization and termination are used.

6.4 1-Phase PWM Signal Generation (PWM)

Asymmetric PWM mode As an alternative to the symmetric PWM generation mode, you can also let each PWM pulse start at the beginning of the corresponding PWM period (asymmetric PWM mode). Switching between symmetric and asymmetric PWM mode applies to all of the four 1-phase PWM output channels. The following illustration shows two active high symmetric and asymmetric 1-phase PWM signals.

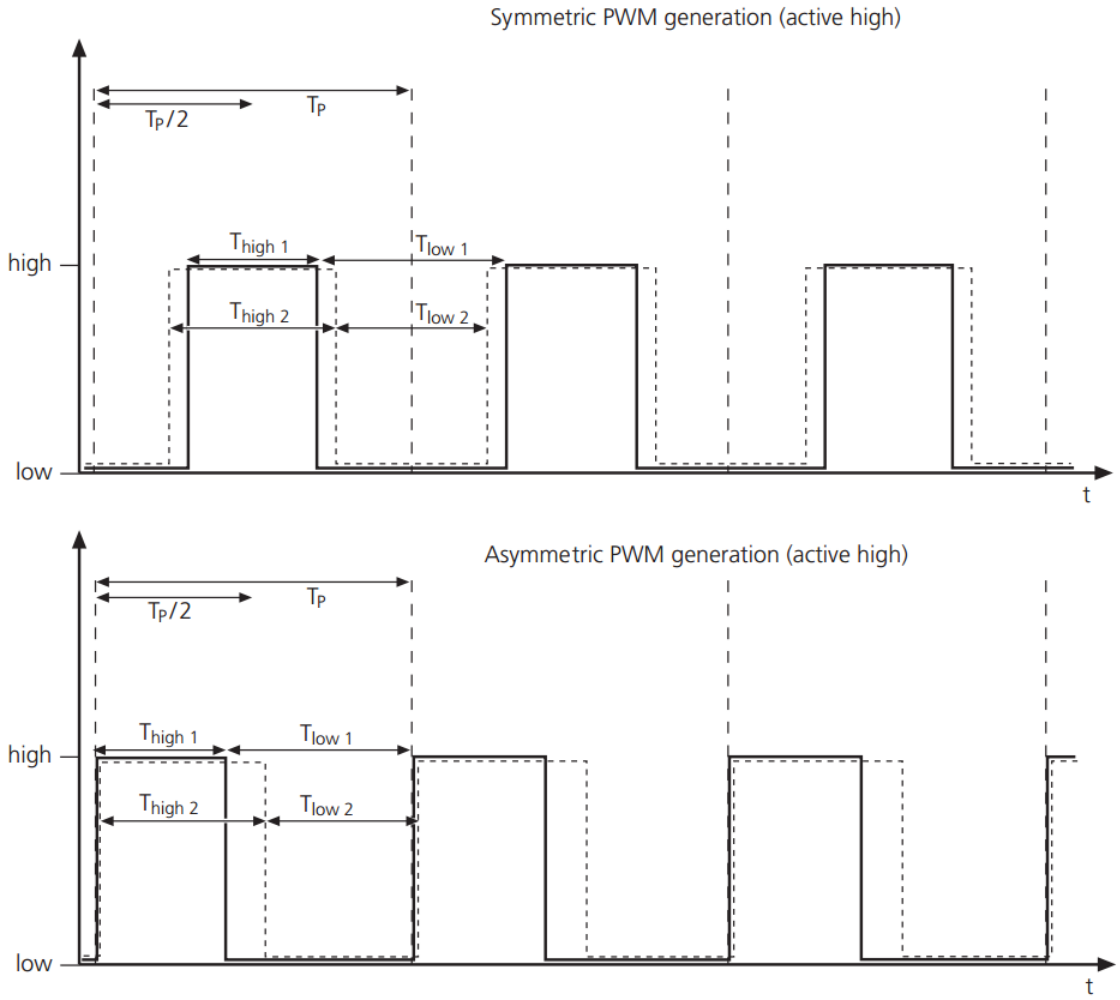

Figure 13: Symmetric and asymmetric PWM generation

PWM period and resolution in asymmetric mode

In the asymmetric mode, the PWM period T_P must be in the range 200 ns ... 409,6 ms. Depending on the period, the following resolutions are given:

Period T_p	Resolution
<3.2 ms	50 ns
<6.4 ms	100 ns
<12.8 ms	200 ns
<25.6 ms	400 ns
<51.2 ms	800 ns
<102.4 ms	1.6 μs
<204.8 ms	3.2 μs
<409.6 ms	6.4 μs

Table 13: Asymmetric PWM resolution

Due to quantization effects, you will encounter considerable deviations between the desired PWM period T_P and the generated PWM period, especially for higher PWM frequencies.

Polarity of PWM signals

For each of the four 1-phase PWM channels, you can specify separately whether to generate active high or active low PWM signals.

PWM output

Via RTI you can specify separately for each of the four 1-phase PWM channels, whether or not to generate PWM signals. In case of PWM stop, the output of each channel can be set to TTL high or low.

RTI/RTLib support

You can perform 1-phase PWM signal generation on the slave DSP via RTI1104 and RTLib1104. For details, see

- DS1104SL_DSP_PWM
- Slave DSP PWM Generation

Using the ST1PWM Pin

Objective The ST1PWM pin is lead to the interrupt controller of the DS1104, so it is possible to generate interrupts that are synchronous to PWM signal generation. You can also use the pin as a further external interrupt input (user interrupt). In this case the ST1PWM pin has to be configured as an input (using RTLib1104).

Strobing I/O

In addition, you can use the ST1PWM pin for strobing the I/O (ADCs, DACs and incremental encoder signals). The required trigger signal can be either generated by the slave DSP or driven externally. In the second case the ST1PWM pin has to be configured as an input or the slave DSP must be in reset mode.

Recognizing signals at the ST1PWM pin

To allow the interrupt controller to recognize an incoming signal at the ST1PWM pin (PWM interrupt, external interrupt or trigger for strobing), the interrupt signal must be kept high for at least 1 μs. The interrupt is activated by the high to low transition of the signal. The signal must remain low for at least 100 ns after the transition.

7 Hardware and software required for this project

To implement a real-time control loop using dSPACE and MATLAB we need following items.

1. dSPACE DS1104 R&D Controller Board

Figure 14: Controller Board

2. Dongle licenses on a USB flash disk

Figure 15: Dongle USB flash

3. Connector panel CLP1104

Figure 16: Connector panel

4. D Sub 37 cable connector

Figure 17: Cable connector

5. Servomotor Futaba S3003

Figure 18: Futaba S3003

6. Ultrasonic sensor HC-SR04

Figure 19: HC-SR04

8 Controller Design and Implementation in Simulink

8.1 Configuration of the Simulink model

1. The first step is start MATLAB and then, the following message appears, which says that dSPACE RealTime Interface (RTI) is installed for several hardware platforms, in this case, select DS1104.

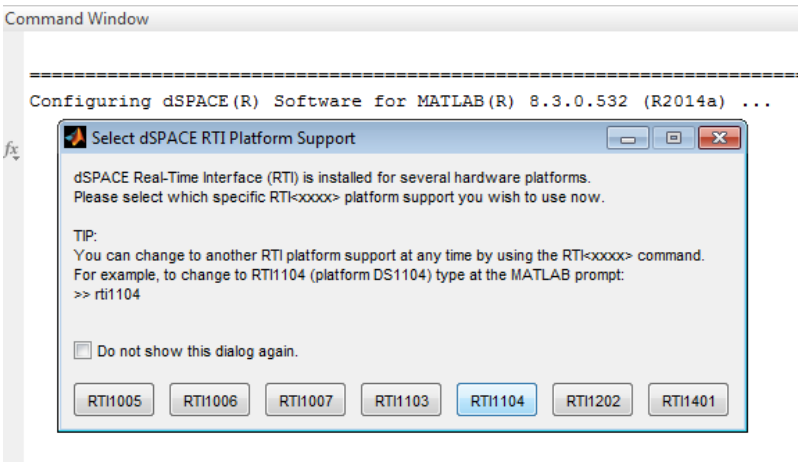

Figure 20: Selection of the RTI platform

By default, the workspace is placed in the user documents. For a better organization of the generated files, change it to another directory because when you compile the Simulink model a **.sdf** file will be generated and without it, you won't be able to work in the ControlDesk platform.

2. Some parameters must be adjusted before importing the model to ControlDesk. First, the solver and step time must be configured. To do so, click on the 'Simulation' button, and click on the option 'Model Configuration Parameters', which will open a new dialog window, as the one seen in figure 21. In 'Solver Options', select the type 'Fixed-step' and select the solver 'ode4 (Runge-Kutta)'. In the option 'Fixed-step size (fundamental sample time)', a value must be entered such as it allows to have a good resolution of data capture, for this project, a value between 0.001 and 0.0001 is admissible.

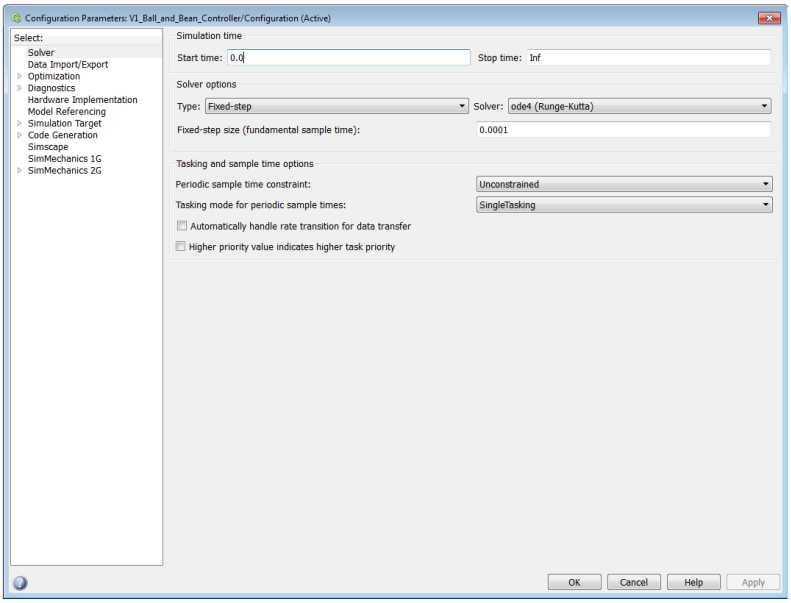

Figure 21: Configuration of integration step

3. The option ‘Simulation Time’ must be set to ‘inf’. This can be done in the option ‘Solver’ of the ‘Model Configuration Parameters’ window, or in the box shown in the toolbar of Simulink, as seen in figure 22. This allows ControlDesk to simulate the model for an indefinite amount of time, instead of stopping the simulation at a specific time, as it is done normally in Simulink.

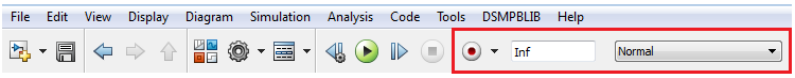

Figure 22: Change of simulation time to ‘Inf’

4. The model must be saved as a .slx file in the main folder defined by the user in step 1.

5. Finally, the model must be compiled in the dSPACE board. The DS1104 board must be selected in the ‘Model Configuration Parameters’ window, selecting the ‘Code Generation’ option, as seen in the figure 23. Press the button ‘Browse. . . ’ at the right of ‘System target file’ to open a window that will show a list of .tlc files, and select the one associated with the DS1104 board, as shown in figure 24.

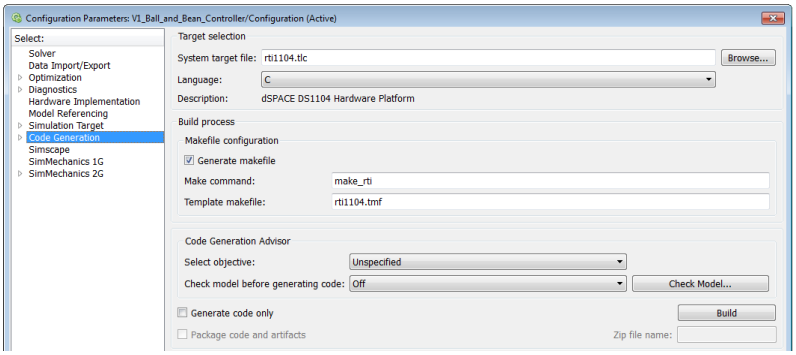

Figure 23: Selection of the board for compilation

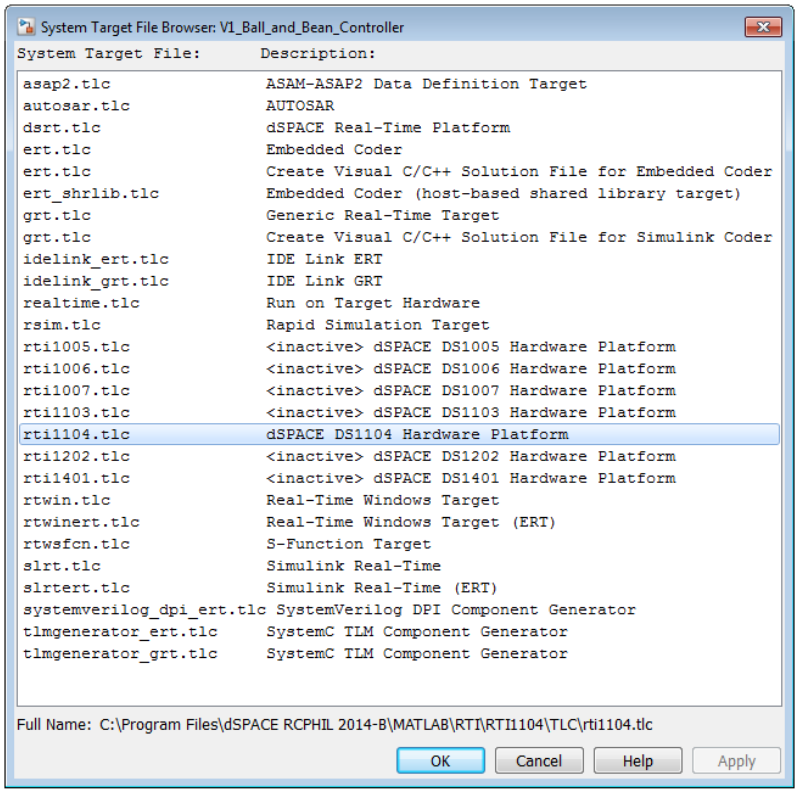

Figure 24: Selection of the .tlc file related to the board

6. The final step is to generate the code that will be compiled in the board. In Simulink, go to the menu 'Code', and in the option 'C/C++ Code' select 'Build model', as seen in figure 25 or press **Ctrl+B**. If it is the first time this process is done after opening MATLAB, the error shown in figure 26 will appear in the 'Diagnostic Viewer' window. To solve the issue, paste the code shown in the message **revertInlineParameterOffTo2013b** in the command window of MATLAB and press enter. The message shown in figure 27 will appear. In the Simulink model, an icon will appear ('RTI Data'), showing that the model is associated with a Real Time Interface, as seen in figure 28. A .sdf file, along with other files and folders will appear in the route selected in MATLAB.

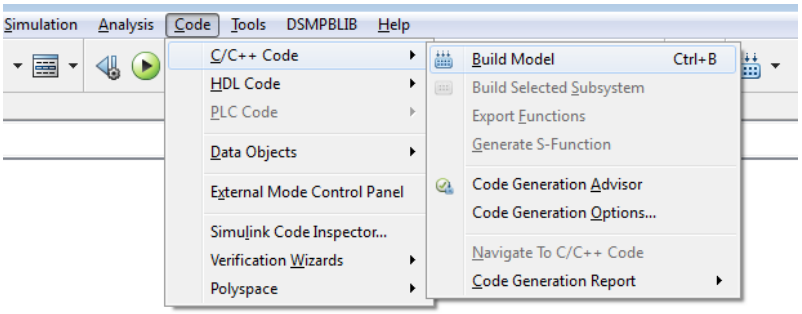

Figure 25: Generation of C code

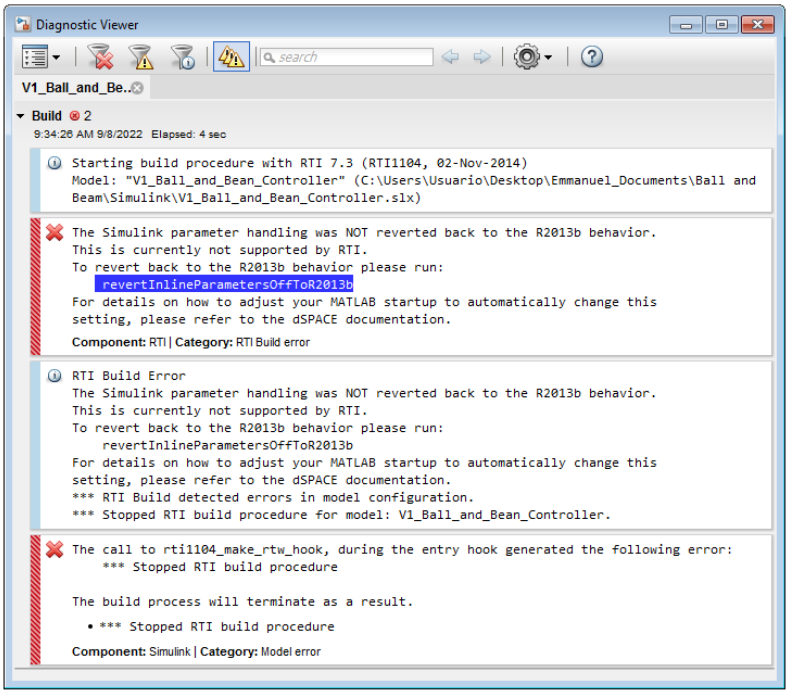

Figure 26: RTI Error message

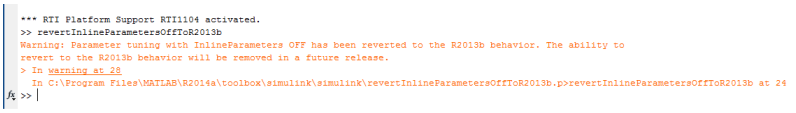

Figure 27: Command window of MATLAB after the correction of the error

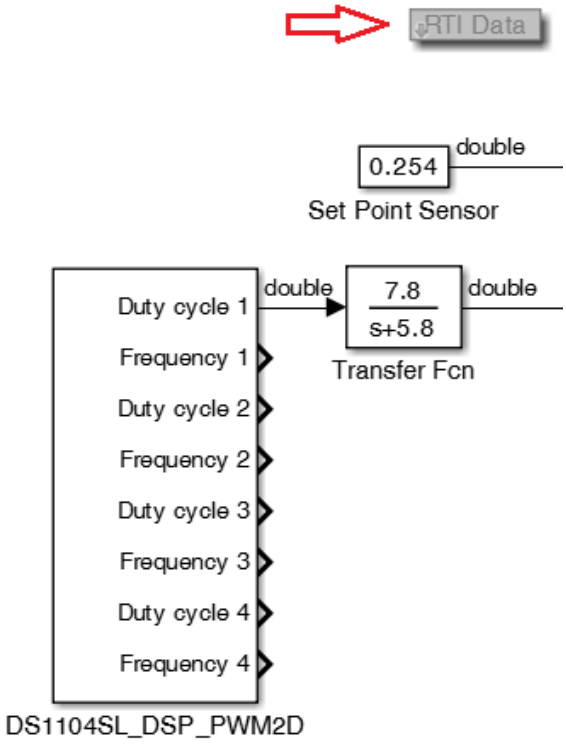

Figure 28: 'RTI Data' symbol in the Simulink model

8.2 dSPACE Simulink blocks

In Simulink, there is available a library of blocks related to the dSPACE board.

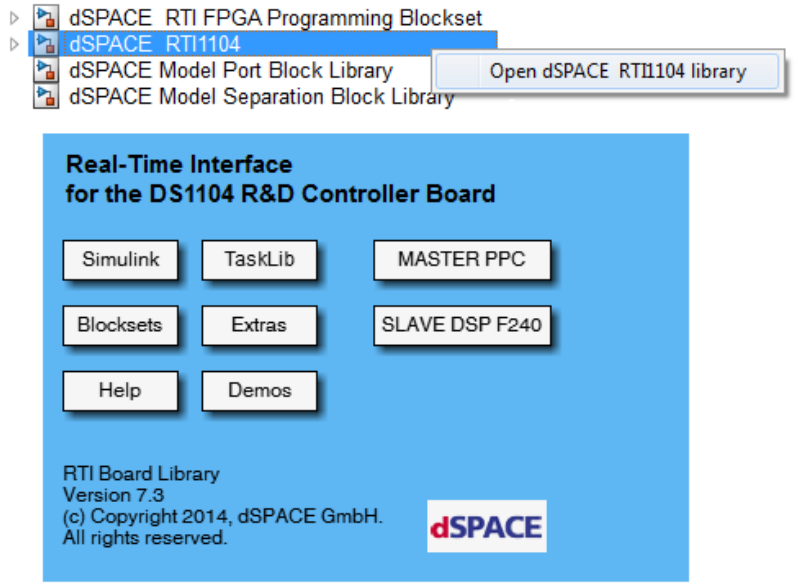

Figure 29: dSPACE libraries in Simulink.

By clicking on the SLAVE DSP F240 button, the window shown in figure 30 be displayed.

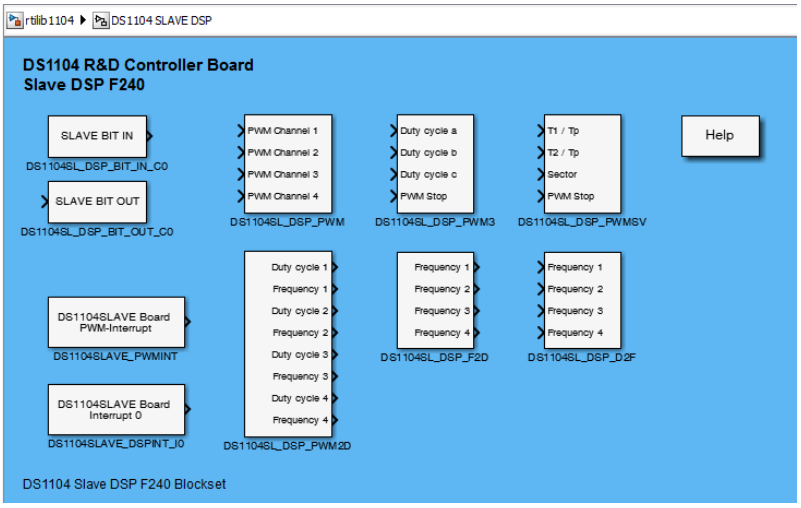

Figure 30: DS1104 Slave DSP F240 Blockset

Here you can find all the blocks related to the DS1104 slave DSP module. For this project only two blocks will be needed:

1. **DS1104SL_DSP_PWM**: for 1 phase PWM generation.
2. **DS1104SL_DSP_PWM2D**: for sensor echo reading.

8.3 Basics of Slave DSP PWM Signal Generation

The slave DSP of the DS1104 provides outputs for PWM signal generation. Each PWM pulse is centered around the middle of the corresponding PWM period (symmetric PWM generation mode) or in phase with the start of the period (asymmetric PWM generation mode).

PWM signals

PWM signal generation is crucial to many motor and motion control applications. PWM signals are pulse trains with fixed frequency and magnitude and variable pulse width. There is one pulse of fixed magnitude in every PWM period.

However, the width of the pulses changes from period to period according to a modulating signal. When a PWM signal is applied to the gate of a power transistor, it causes the turn-on/turn-off intervals of the transistor to change from one PWM period to another, according to the same modulating signal. The frequency of a PWM signal is usually much higher than that of the modulating signal, or the fundamental frequency, so that the energy delivered to the motor and its load depends mainly on the modulating signal.

PWM period, duty cycle and resolution

For PWM signals, you can specify the PWM period TP (= T_high+ T_low) in the range 200 ns . . . 819,2 ms. For 1-phase PWM signals, the PWM period TP applies to each of the four PWM output channels.

You can also specify the duty cycle. The following illustration shows how the duty cycle d (= T_high/ TP) is defined. The available duty cycle range is 0 . . . 1 (0 . . . 100 %).

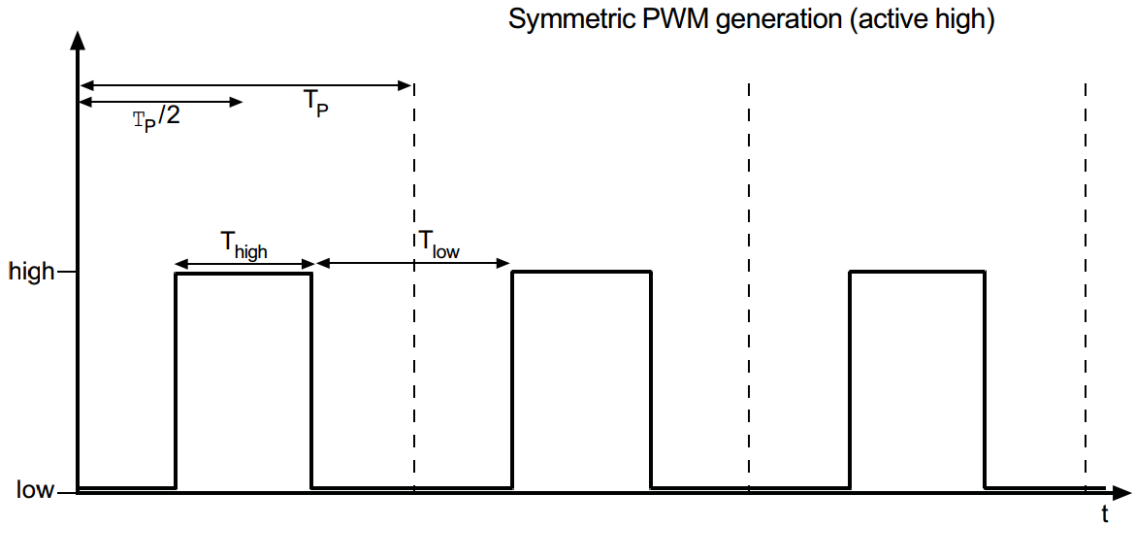

Figure 31: Symmetric PWM generation

Depending on the selected PWM period, the following resolutions are given. They apply to symmetric PWM signals.

Period T_p	Resolution
<6.4 ms	100 ns
<12.8 ms	200 ns
<25.6 ms	400 ns
<51.2 ms	800 ns
<102.4 ms	1.6 μs
<204.8 ms	3.2 μs
<409.6 ms	6.4 μs
<819.2 ms	12.8 μs

Table 14: Symmetric PWM resolution

Asymmetric PWM mode

As an alternative to the symmetric PWM generation mode, you can also let each PWM pulse start at the beginning of the corresponding PWM period (asymmetric PWM mode). Switching between symmetric and asymmetric PWM mode applies to all of the four 1-phase PWM output channels. The following illustration shows two active high symmetric and asymmetric 1-phase PWM signals.

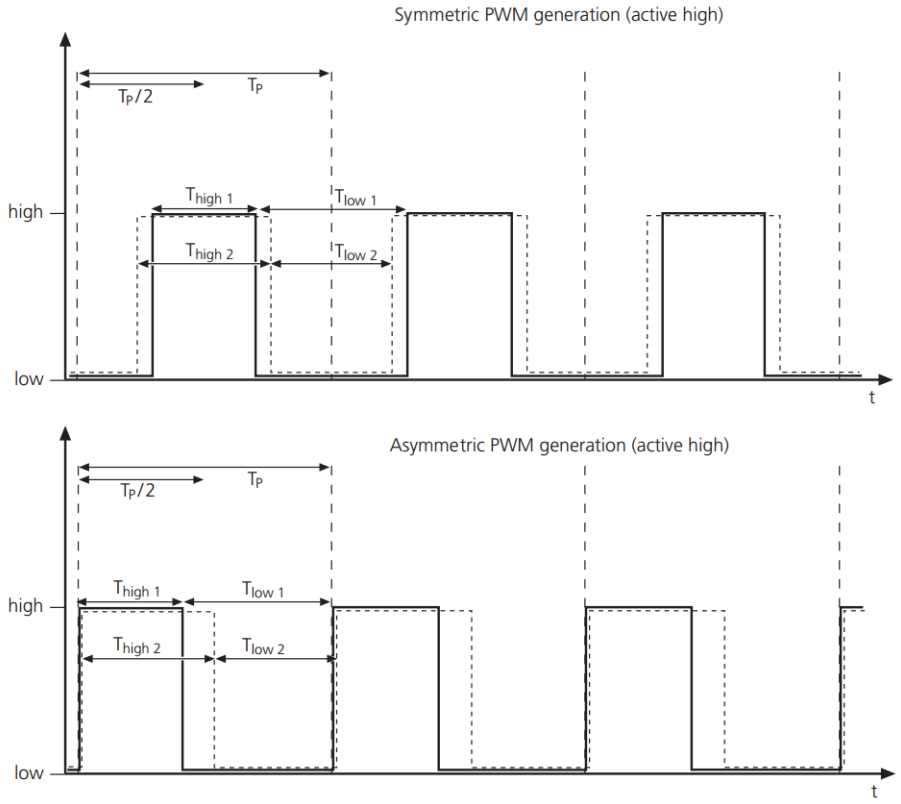

Figure 32: Symmetric and asymmetric PWM generation

PWM period and resolution in asymmetric mode

In the asymmetric mode, the PWM period T_P must be in the range 200 ns ... 409,6 ms. Depending on the period, the following resolutions are given:

Period T_p	Resolution
<3.2 ms	50 ns
<6.4 ms	100 ns
<12.8 ms	200 ns
<25.6 ms	400 ns
<51.2 ms	800 ns
<102.4 ms	1.6 μs
<204.8 ms	3.2 μs
<409.6 ms	6.4 μs

Table 15: Asymmetric PWM resolution

Due to quantization effects, you will encounter considerable deviations between the desired PWM period T_P and the generated PWM period, especially for higher PWM frequencies.

PWM outputs

The PWM outputs can be specified. The running PWM generation can be suspended and the corresponding channels can be set to a specified TTL level (high or low). Only the output of the PWM signal is disabled. Signal calculation is still running and if you enable PWM generation, the currently calculated signal is output, and not the defined initialization or termination value. The PWM outputs can be specified for the two simulation phases (RTI):

- During the initialization phase, you can disable the PWM generation of selected channels and set each output (pair) to a defined TTL level (high or low). No signal is generated during the initialization.
- During run time, you can stop PWM generation and set the outputs to a defined TTL level (high or low). At any time you can resume in generating the PWM signal. If the simulation terminates the outputs can be set to defined TTL levels.

If the PWM stop feature is disabled, the normal initialization and termination routines are executed. That means the specified duty cycles for initialization and termination are used.

8.4 1-Phase PWM Signal Generation - DS1104SL_DSP_PWM

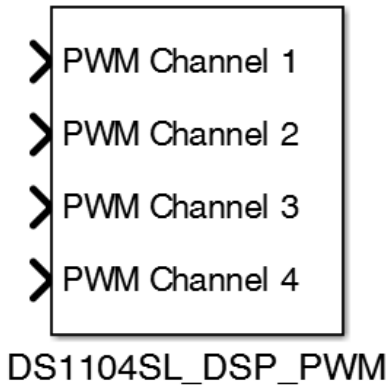

Figure 33: DS1104SL_DSP_PWM module in Simulink

Purpose

To generate standard PWM signals with variable duty cycles and enable PWM stop during run time.

Description

For 1-phase PWM generation, a PWM stop can be specified to suspend PWM signal output during run time. The outputs of the channels are set to a defined TTL level. The dimensions of the inports are set to 2, which allows you to enter two values over the same port. This can be done via a Simulink MUX block, for example. Value 1 specifies the duty cycle and value 2 the PWM stop behavior. If you set value 2 to "0" a PWM signal is generated, "1" suspends signal generation and sets the output to the specified TTL level. If the PWM stop is disabled for a channel only the duty cycle can be input. Although you can disable the PWM stop feature for each channel during run time, you can specify whether you want to set the PWM output to a specified TTL level or to generate a signal during the initialation phase.

I/O mapping

Signal	Channel/Bit Numbers of Related RTI Blocks/RTLib Functions			I/O Pin on ...
	Related RTI Block(s)	Ch/Bit (RTI)	Ch/Bit (RTILib)	D-SUB 37
1-Phase PWM Signal Generation (PWM)				
• TTL output voltage range • Output current range: ±13 mA				
ST2PWM	DS1104SL_DSP_PWM	Channel 1	Channel 1	5
SPWM7		Channel 2	Channel 2	10
SPWM8		Channel 3	Channel 3	29
SPWM9		Channel 4	Channel 4	11

Table 16: Slave DSP 1-phase PWM Signal

I/O characteristics

Simulink Inport	Input	Value	Data Type	Meaning
PWM Channel 1 ... 4	Duty cycle 1 ... 4	0 ... 1	Double	Duty cycle of the PWM signal for channel 1 ... 4
	PWM Stop 1 ... 4	0 ... 1	Boolean	Enables PWM stop for channel 1 ... 4: • Value 1 stops PWM generation • Value 0 resumes PWM generation

Table 17: 1-phase PWM Signal Simulink data types

8.4.1 Simulink implementation

The DS1104SL_DSP_PWM inputs PWM Channel 1 ... 3 were used and channel 4 was disabled.

- **PWM Channel 1**, is used for pulse generation for triggering the HC-SR04 ultrasonic sensor.
- **PWM Channel 2**, is used to command the servo motor linked to the sensor and the ball guide.
- **PWM channel 3**, is used to lock the two servo motors of the base. Just to make sure that nothing will move unintentionally.

- **PWM channel 4**, disabled.

The block was configured to operate at **50 Hz** in **asymmetric mode**. All other parameters were left at their default values. Like mentioned above, all four inputs were multiplexed with a **STOP bit** to provide the start/stop option.

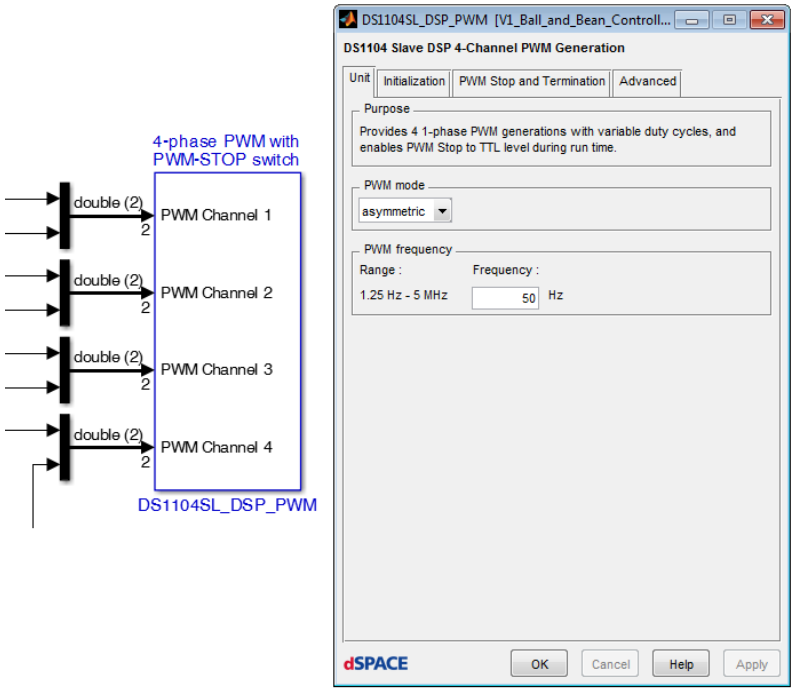

Figure 34: PWM module in Simulink

8.5 1-Phase PWM Signal Measurement - DS1104SL_DSP_PWM2D

In this case the DS1104SL_DSP_PWM2D block is used.

It can measure the duty cycles and the frequencies from up to 4 independent channels used for PWM-type signals.
The reason for its use is because the HC-SR04 returns a high pulse proportional to the duration of the echo of the sound emitted by the sensor actuator.

The measurement algorithm used is accurate if the PWM period starts with the falling or rising edge of the corresponding PWM signal so, the PWM module must be in **asymmetric mode**.

As only one signal will be measured, channel 1 is used to measure its duty cycle.

Signal	Channel/Bit Numbers of Related RTI Blocks/RTLib Functions			I/O Pin on ...
	Related RTI Block(s)	Ch/Bit (RTI)	Ch/Bit (RTILib)	D-SUB 37
Slave DSP PWM Signal Measurement (PWM2D)				
• TTL input voltage range 0 - 5V • Input current: 500 μA				
SCAP1	DS1104SL_DSP_PWM2D	Channel 1	Channel 1	2
SCAP2		Channel 2	Channel 2	21
SCAP3		Channel 3	Channel 3	3
SCAP4		Channel 4	Channel 4	22

Table 18: Signal mapping of Slave DSP PWM Signal Measurement (PWM2D)

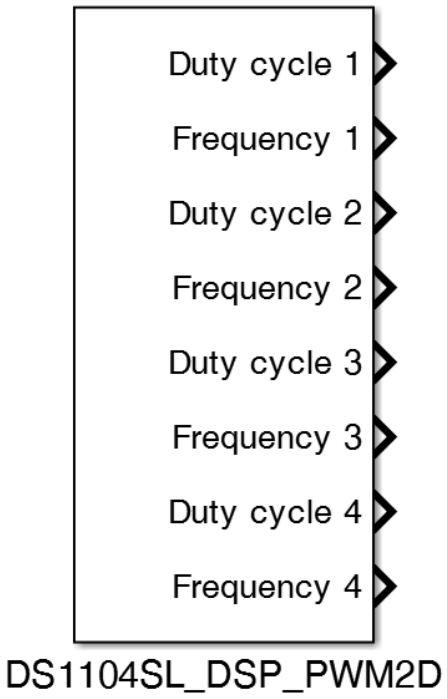

Figure 35: Measurement module in Simulink

8.6 Basic Operation and Timing of the HC-SRO4 Ultrasonic Sensor

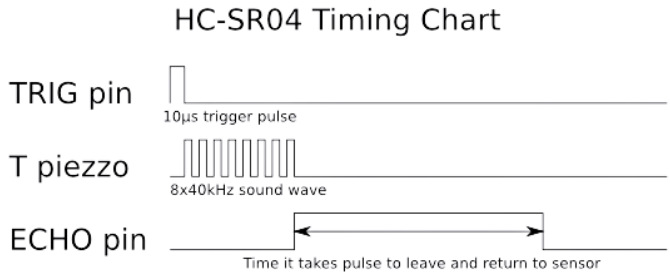

Figure 36: HC-SR04 Timing chart

1. Make "Trig" (pin 2) on the sensor high for 10μs. This initiates a sensor cycle.
2. 8x40kHz pulses will be sent from the "T" transmitting piezzo transducer of the sensor, after which time the "Echo" pin on the sensor will go from low to high.
3. The 40kHz sound wave will bounce off the nearest object and return to the sensor.
4. When the sensor detects the reflected sound wave, the the Echo pin will go low again.
5. The distance between the sensor and the detected object can be calculated based on the length of time the Echo pin is high.
6. If no object is detected, the Echo pin will stay high for 38ms and then go low.

8.7 Basic Operation and Timing of Futaba S3003 Servomotor

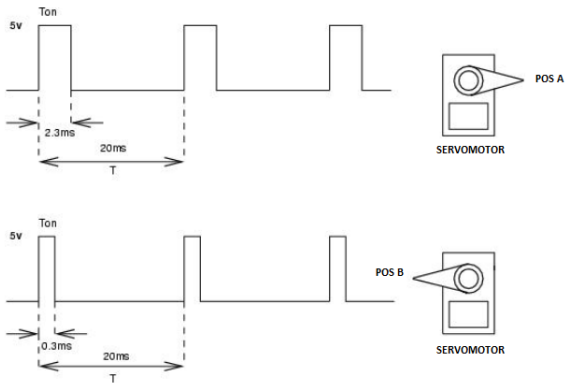

Figure 37: Servomotor Timing chart

To position the servo, a periodic signal of 50Hz (20ms period) must be applied. The width of the pulse determines the servo position. If the width is 2.3ms, the servo is positioned at one end and if the width is 0.3ms it is positioned at the opposite end. Any other width between 0.3 and 2.3ms places the servo in a position between one end and the other. For example, if we want it to be positioned exactly in the center, we apply a width of 1.3ms.

When the signal is no longer sent, the servo enters an idle state, and can therefore be moved by hand. As long as the signal is applied, it will remain fixed in its position, forcing it to stay there.

8.8 Finding the limits of the servomotor - Simulink example model

Simulink has the simplicity of block programming and the programmer must have the skills to realize a simple, efficient and robust control system.

You must know how your servo motor operate and especially its limits. You can build a simple model in Simulink and find these limits with ControlDesk. Not all motors are the same and to be accurate, this is a necessary step.

For this example a *20ms* period will be set, that is the typical value for this type of servomotors (this value also change whit the servomotor model).

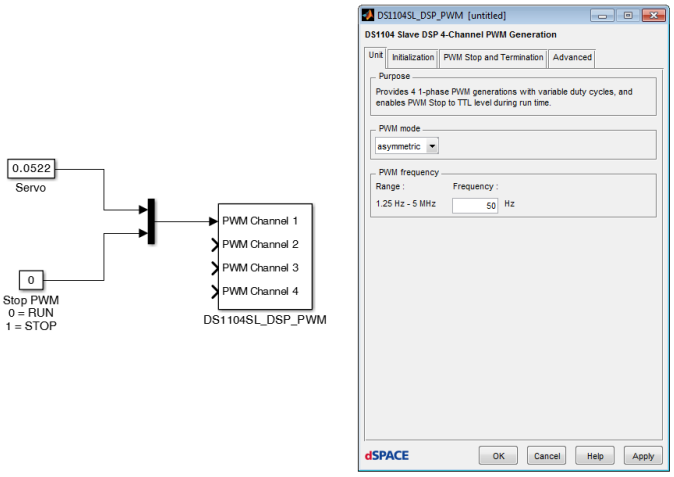

Figure 38: Simulink servomotor test

You must now follow the compilation steps mentioned in the first section in order to work with ControDesk. After compilation process, the RTI Data symbol will appear.

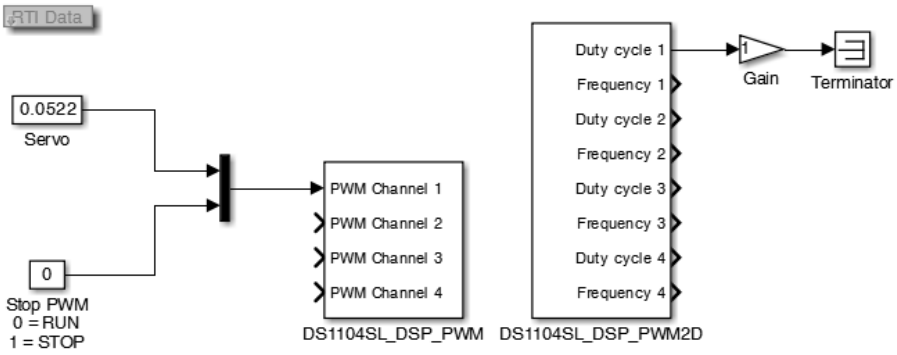

Figure 39: Simulink model - servomotor test

With this model you will be able to test your servomotors to know their characteristics and operating limits.

Remember that most servomotors operate between 0,3 and 2,3 ms pulse width for their entire range of motion.

8.9 ControlDesk interface

8.9.1 Creating a ControDesk project

1. Open ControlDesk 5.3. The first thing to do is to create a new project. This is done in the 'File' tab, in the option 'New', and selecting 'Create New Project and Experiment', as seen in figure 40.

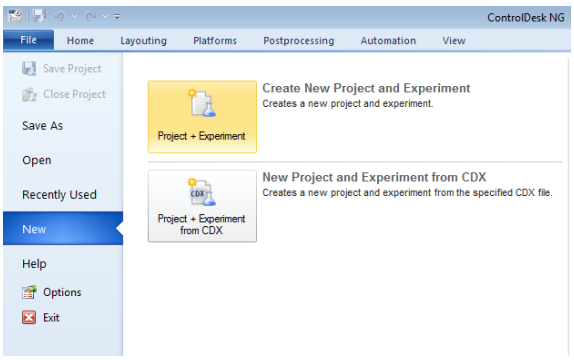

Figure 40: Creation of a new project in ControlDesk

2. A window will open. Select a name for the project and a root directory.

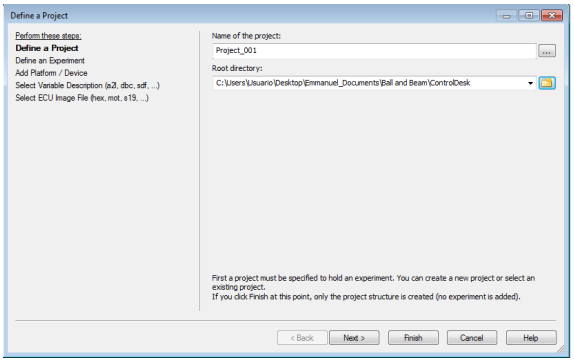

Figure 41: Define a project

3. The root directory must be in you project main folder. Just follow the steps in figure 42.

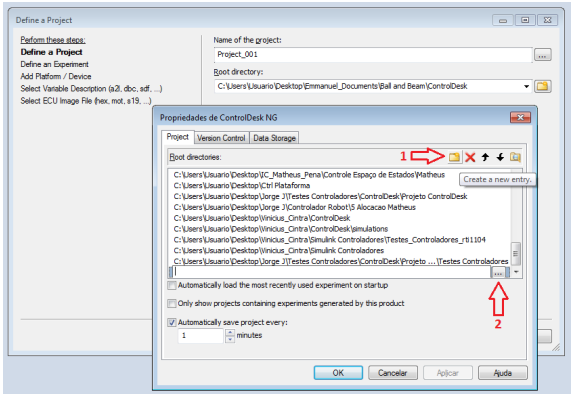

Figure 42: Root directory

4. After pressing ‘Next’, the user must select the controller board which is being used which, in this case, is the DS1104, as seen in figure 43.

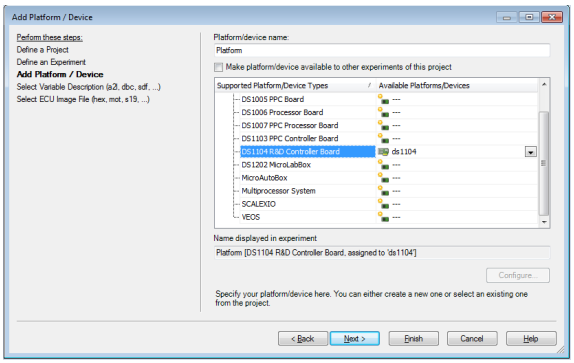

Figure 43: Add plataform / Device

5. To associate the variables from the Simulink model with ControlDesk, the **.sdf** file generated during the compilation of the code done previously must be imported by pressing the button ‘Import from file...’, and selecting it, as seen in figure 44. After this is done, press ‘Finish’.

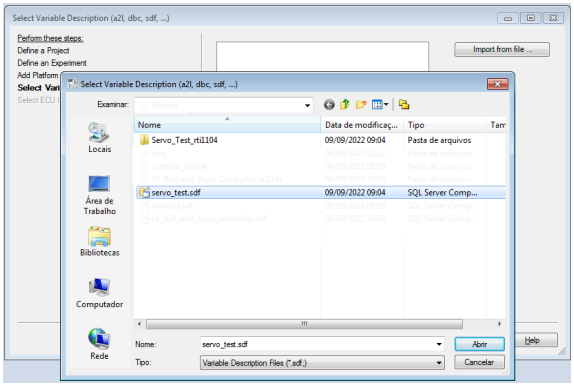

Figure 44: Select variable description

6. The software offers a variety of tools and options. The ones of interest are highlighted with colors in figure 45, are described below:

   - Yellow area: corresponds to the lead toolbar.
   - Red area: corresponds to the work area. In this zone, a variety of instruments like indicators, graphs, and numeric inputs can be placed, to create a ‘Layout’, which acts as a interface between the user and the controller.
   - Blue area: the instrument selector shows all the instruments available. If this list does not appear at the moment the project is created, it can be activated in the menu ‘Layouting’, and selecting ‘Insert Instrument’.
   - Green area: shows the variables associated with the imported Simulink model, or models. The instruments in the layout can be associated to variables of the Simulink model.

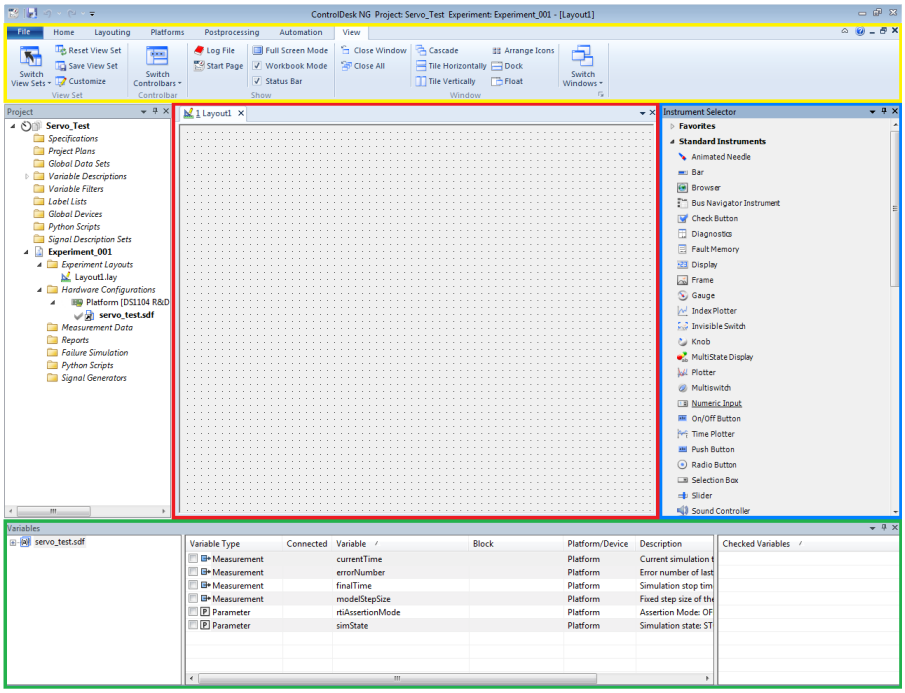

Figure 45: ControlDesk window

8.9.2 Data recording

1. To be able to work with historical data of the variables, it is necessary to record their behavior during a real time experiment, and export them to a file. To analyze the data in MATLAB, a file can be generated with the recorder tool of ControlDesk. To display the 'Measurement Configuration' bar, go to the tab 'View', click on 'Switch Controlbars' and select 'Measurement Configuration', as seen in figure 46

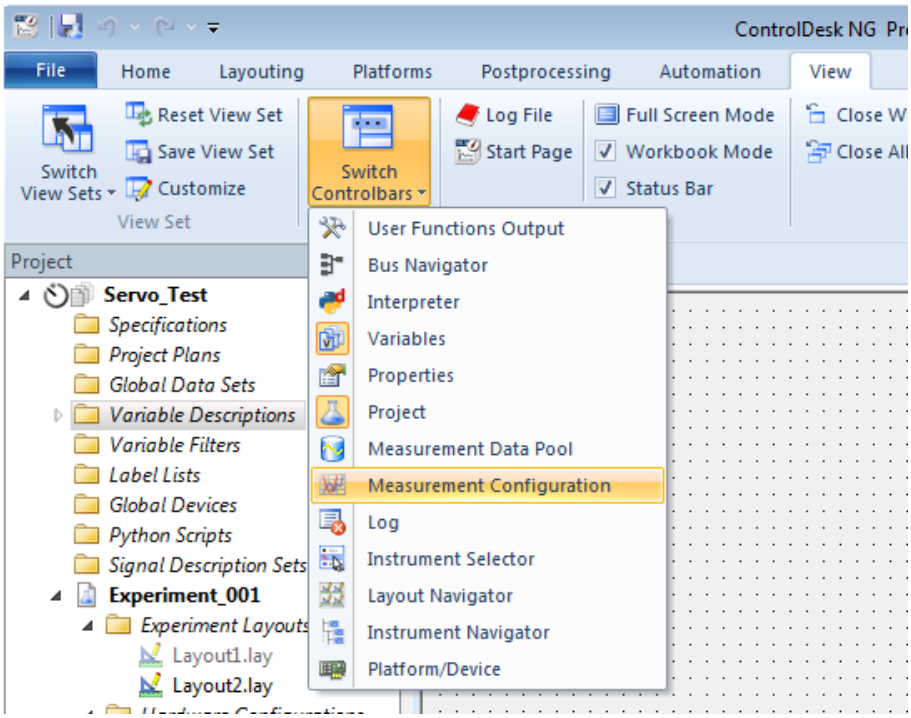

Figure 46: Activation of the recording tool bar

2. In the ‘Measurement Configuration’ bar one can find the ‘Recorders’. ControlDesk records the variables shown in the ‘Plotters’ of the layout by default, so the ‘Recorder 1’ corresponds to the actives plotters. To record another variable, one can drag and drop a variable from the ‘Variables’ bar to a existing recorder, or create one by right-clicking on ‘Recorders’, and selecting ‘Create New Recorder’.

3. Right-click on ‘Recorder 1’ and select ‘Properties’ to open the dialog seen in figure 47. Select ‘Automatic export’ to automatically create a file with the recording of the variable, and select a name for the file, the folder where the file will be generated, and the file type. To be able to work the data in MATLAB select ‘MATLAB file (*.mat)’. Also, the name of the recorder can be changed

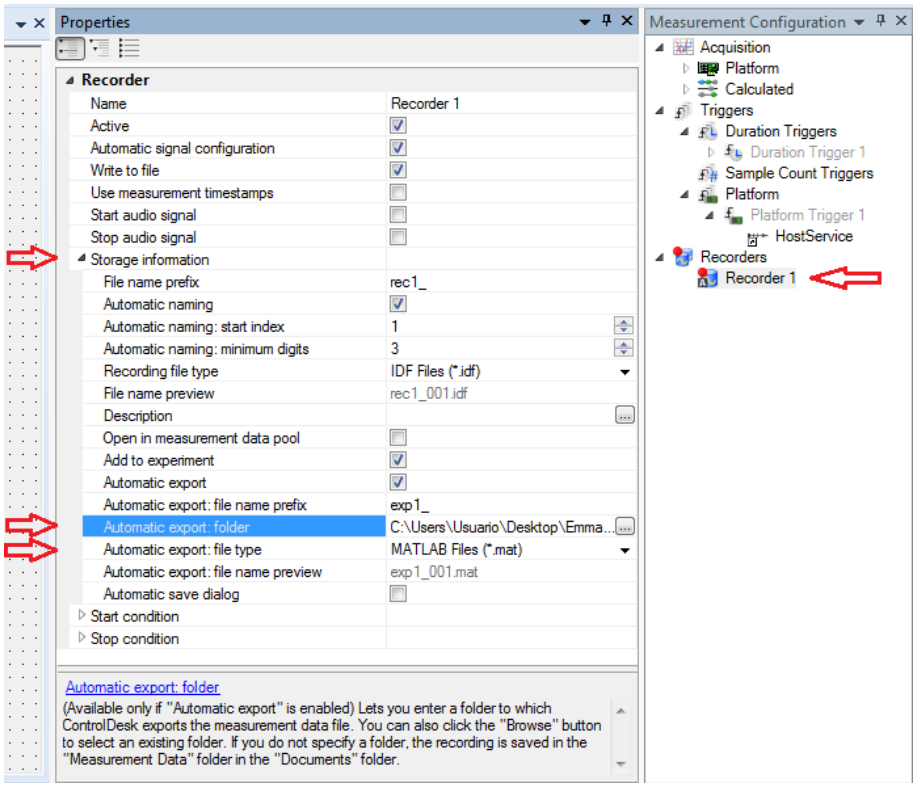

Figure 47: Properties of the recorder

4. To record the data, in the ‘Home’ tab, select ‘Start Immediate’ when the experiment is running, or any other time to initiate the experiment and record from the beginning. Also, the recording can be set to start automatically in response to a variable value or a logical expression, by selecting ‘Trigger Rules’.

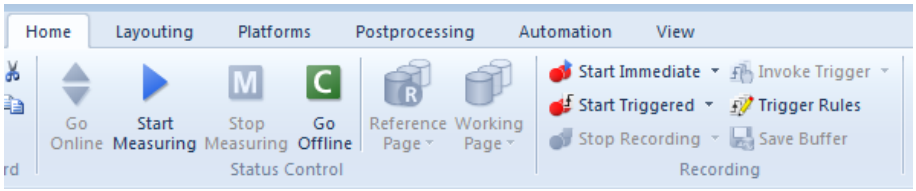

Figure 48: Start recording

8.9.3 Layout interface

The layout area corresponds to the zone were a variety of instruments like indicators, graphs, and numeric inputs can be placed, to create a 'Layout', which acts as a interface between the user and the controller. For this example we only need a few elements:

1. Numeric inputs: To work with the servo motor PWM and to START/STOP the PWM output.
2. Numeric display: To visualize the value of the generated PWM.
3. Time plotter: To visualize the value of the generated PWM over the time.

To place components 1, 2 and 3 in the layout area, you have to drag and drop the corresponding variables from the **variables area** and then associate them with the desired instrument.

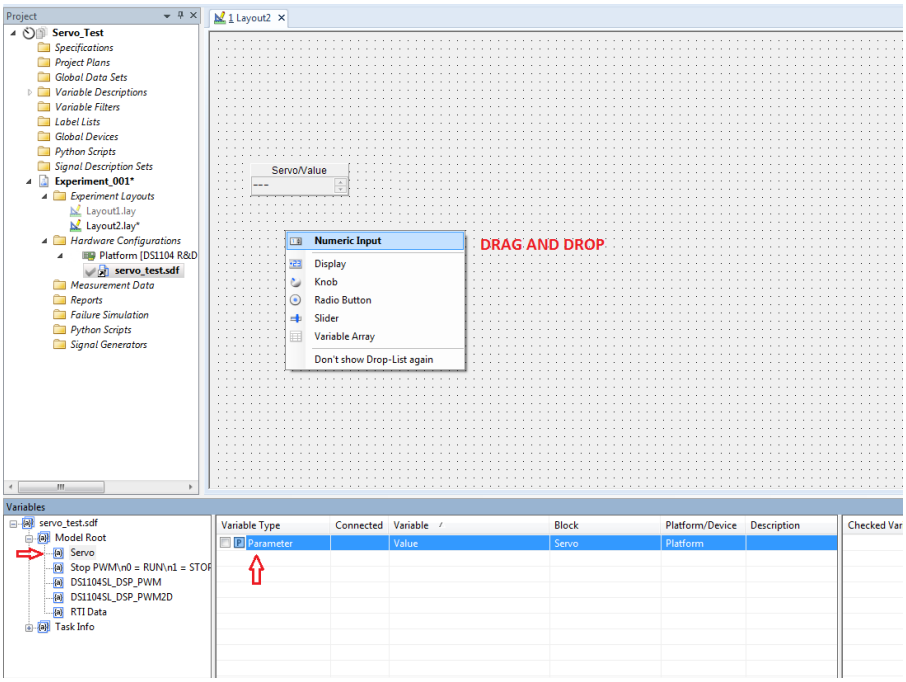

Figure 49: Servo numeric input

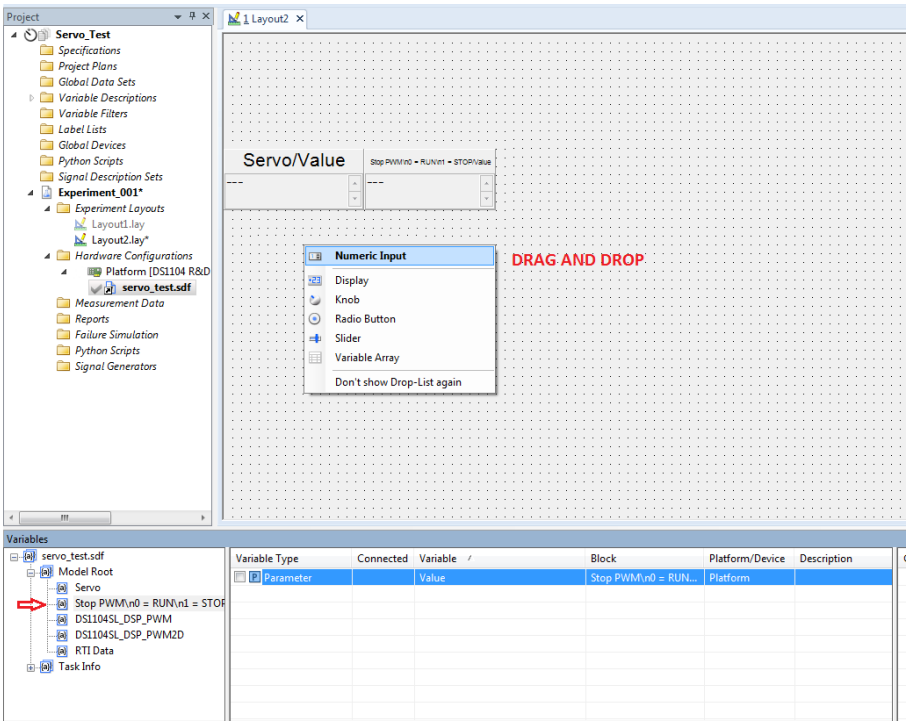

Figure 50: START/STOP numeric input

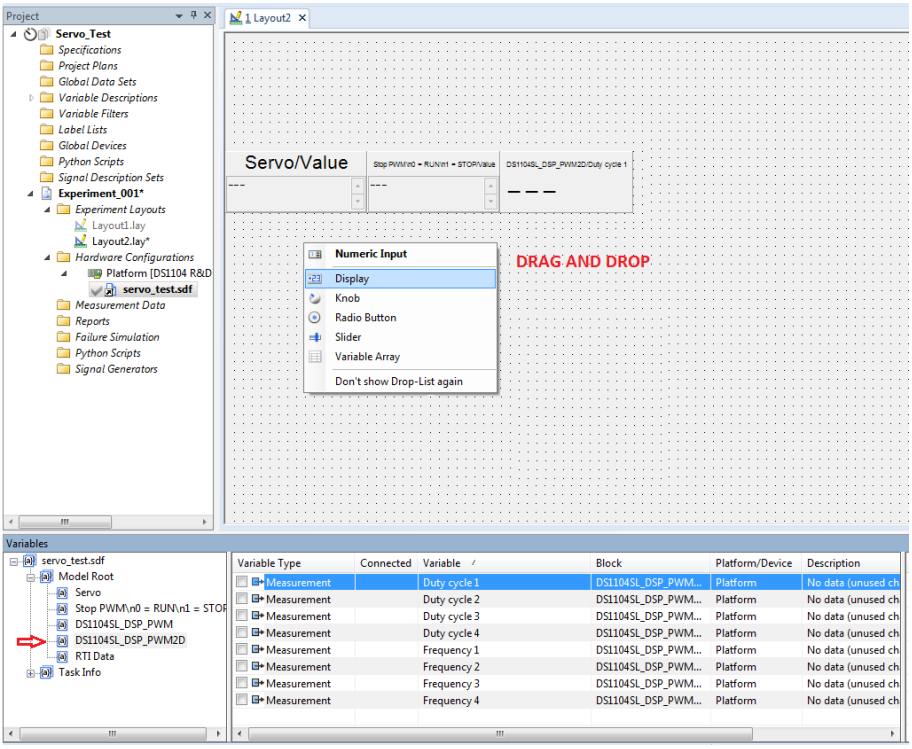

Figure 51: PWM numeric display

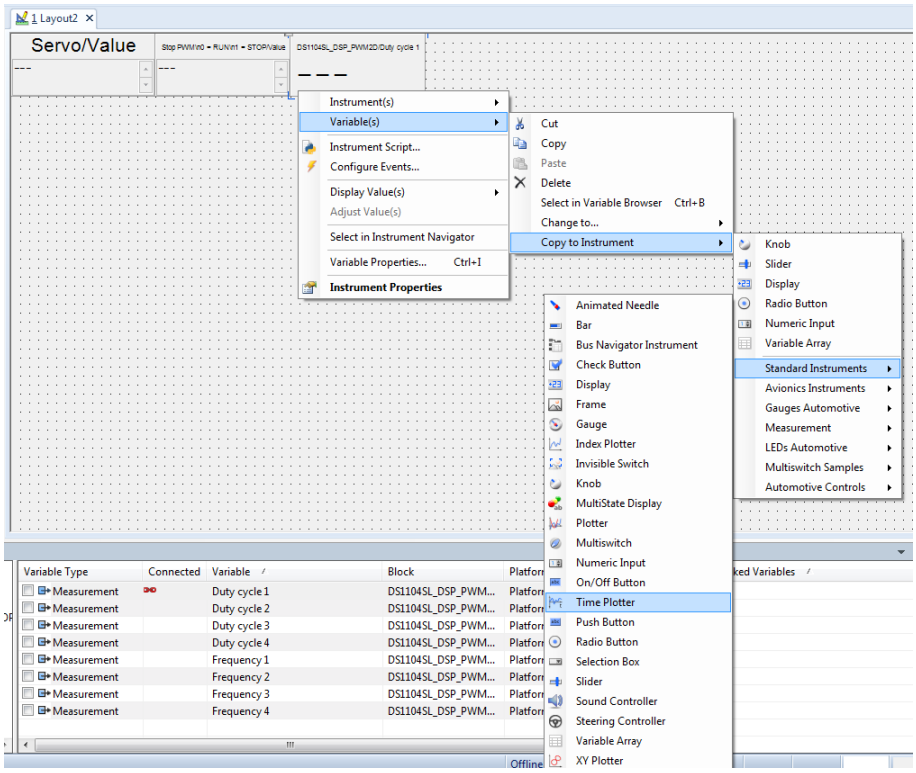

Figure 52: Adding plotter linked to PWM duty cycle

With these items, you can now press the **Go Online** and **Start Measuring** buttons to test the experiment.

9 Ball and beam software implementation

The objective of this project is the practical implementation of a ball and beam system using the dSPACE digital control system with a servomotor, as actuator, and an ultrasonic sensor, as an element to measure error. Using the DS1104 Slave DSP F240 Blockset and the above mentioned blocks will help in these tasks.

10 KiCAD

11 Ball and beam hardware implementation

References

[1] RT-HIL Academy. *Real-Time Hardware-In-The-Loop Simulation: Fundamentals & Popular Platforms*. URL: https://www.youtube.com/watch?v=jcpL6gjT-nk. (accessed: 12.08.2022).

[2] Darko Hercog and Karel Jezernik. "Rapid control prototyping using MATLAB/Simulink and a DSP-based motor controller". In: *International Journal of Engineering Education* 21.4 (2005), p. 596.

Appendix A

Appendix B