MathWorks

MathWorks
Accelerating the pace of engineering and science

4月 - 北京 · 上海 · 深圳

2015 MATLAB 巡回研讨会

技术融合的时代

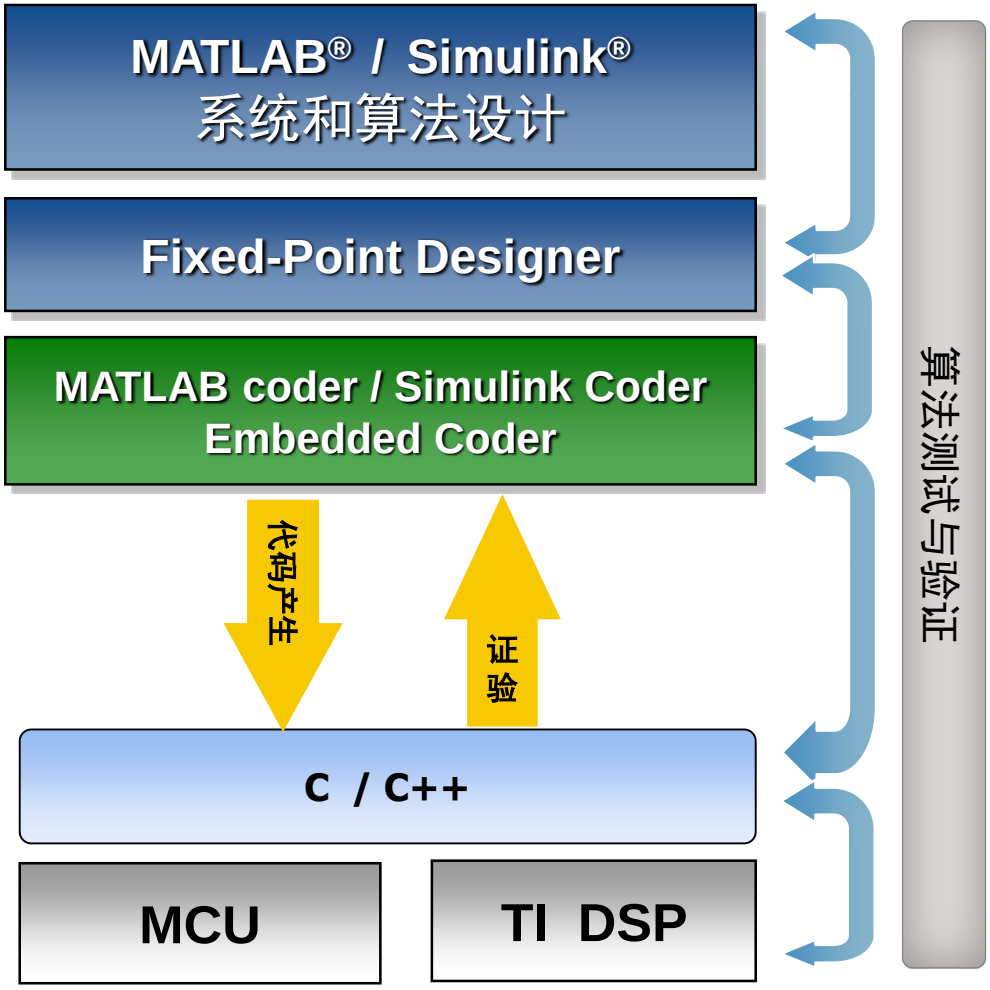
运用MATLAB加速嵌入式算法开发

徐正高
MathWorks 中国
高级项目工程师

主要内容

- 国内典型用户案例分析 - TI DSP系统的实现

基于模型的设计


MATLAB® / Simulink®
系统和算法设计
Fixed-Point Designer
MATLAB coder / Simulink Coder
Embedded Coder
代码产生
证验
C / C++
MCU
TI DSP
算法测试与验证

某项目案例：

基于模型的设计，从MATLAB算法到DSP产品化

1. MATLAB算法

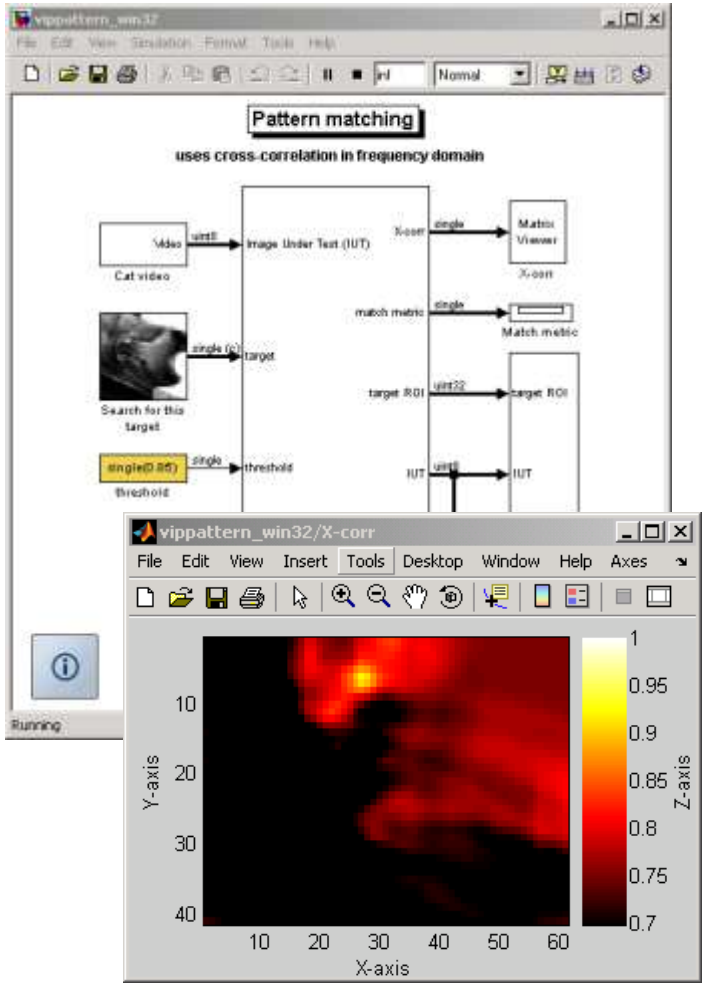
2. Simulink模型
   - 仿真
   - 基于帧的转换
   - 定点化

3. 自动代码产生
   - 快速原型
   - 面向特定芯片的优化
   - 自定义模块

4. 验证
   - 仿真验证

MATLAB Coder \ Simulink Coder
自动产生C代码

- 从视频和图像处理模型产生ANSI-C代码
  - 建立算法和系统的快速原型
  - 在目标硬件上进行测试和验证
  - 方便移植到不同的目标处理器

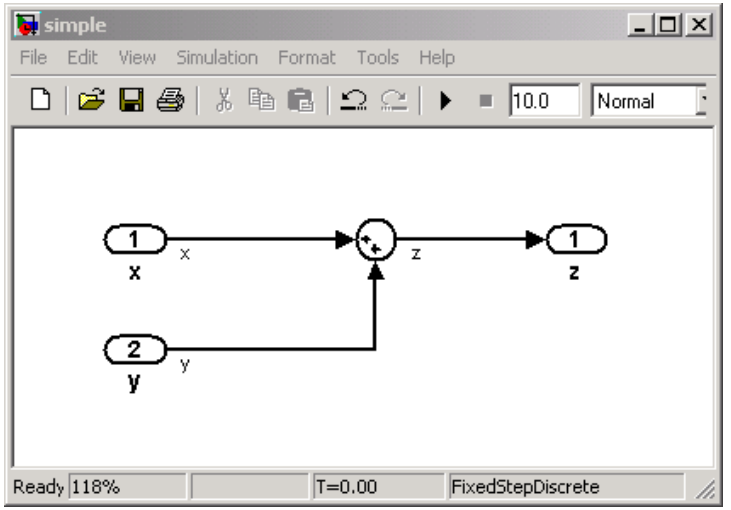


产生代码实例

Rapid Prototype Code (grt.tlc)

```
14  /* External outputs (root outports fed by si
15  ExternalOutputs_simple_grt simple_grt_Y;
16
17  /* Real-time model */
18  RT_MODEL_simple_grt simple_grt_M_;
19  RT_MODEL_simple_grt *simple_grt_M = &simple_
20
21  /* Model output function */
22  static void simple_grt_output(int_T tid)
23  {
24    /* Outport: '<Root>/z' incorporates:
25     *  Inport: '<Root>/x'
26     *  Inport: '<Root>/y'
27     *  Sum: '<Root>/Sum'
28     */
29    simple_grt_Y.z_c = x
30    UNUSED_PARAMETER(tid)
31  }
```

```
119  /*=========================================
120   * Start of GRT compatible call interfa
121   *=========================================
122  void MdlOutputs(int_T tid)
123  {
124    simple_grt_output(tid);
125  }
126
127  void MdlUpdate(int_T tid)
128  {
129    simple_grt_update(tid);
130  }
131
132  void MdlInitializeSizes(void)
133  {
```



Production code (ert.tlc)

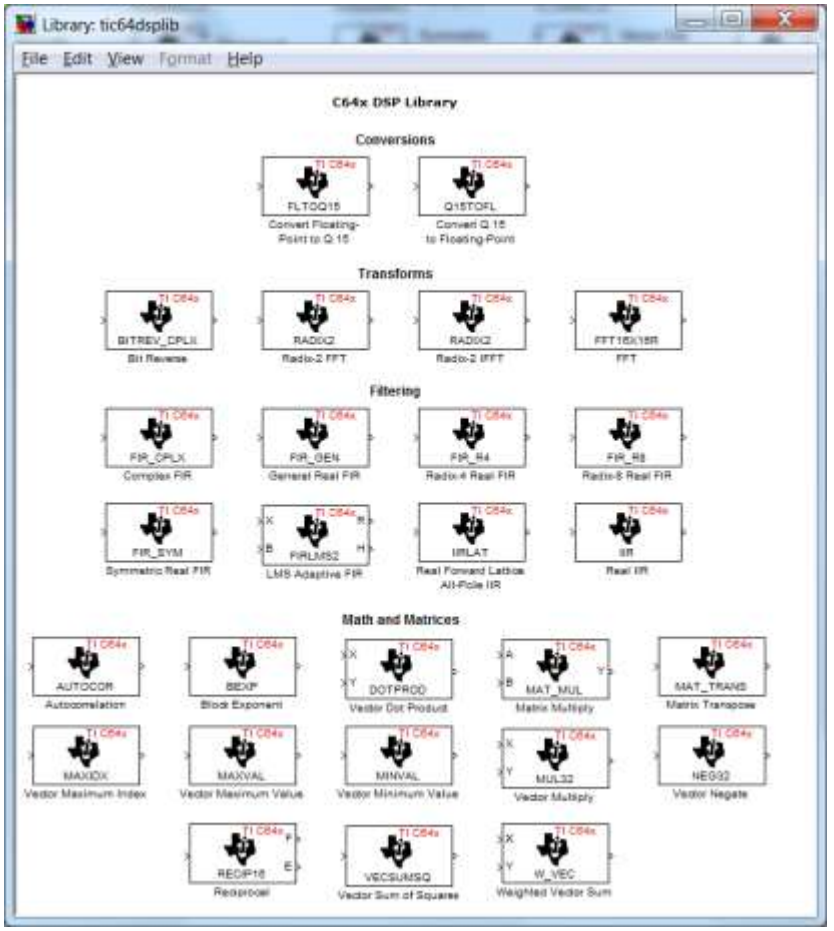
```
1  #include "simple_ert.h"
2  #include "simple_ert_private.h"
3
4  real_T z;
5  void simple_ert_step(void)
6  {
7    z = x + y;
8  }
9
10 void simple_ert_initialize(void)
11 {
12 }
```

Embedded coder

- 产生产品级代码
  - 专用处理器，优化代码
  - **Simulink 模块和优化库(FIR, FFT, …)**
- 工程自动化
  - 自动产生整个工程
  - **CCS、编译器、连接器的API函数**
- 快速原型
  - 特定目标和集成环境
  - **Simulink硬件模块，支持设备驱动（ADC, DAC, RTDX, 子卡)**
- 支持
  - 开发板: **TI 6713 DSK, C6416DSK, DM642 EVM,C6455EVM, DM6437EVM, DM648EVM, C6747EVM,**
  - **DSPs: TI DM64x, C64x, C62x, and C67x families**

自动代码产生

- C语言: DSP
  - 标准C语言(ANSI C / ISO C)
    - MATLAB coder
    - Simulink coder
  - 优化的目标代码
    - Embedded coder

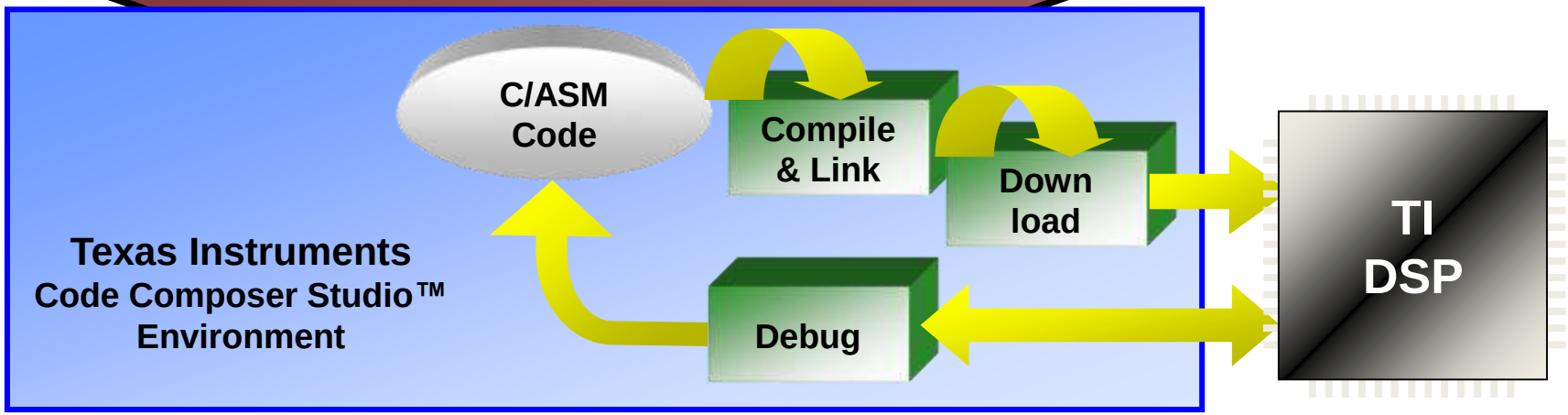


基于模型的设计应用于TI DSPs 及产品

- Model and Simulate in Floating- and Fixed-Point
- Generate embeddable C-code from Simulink and / or Embedded MATLAB
- Rapidly Prototype and Implement on TI processors and boards

MATLAB® Simulink® Stateflow®

- Link for use with Code Composer Studio
- Embedded Coder
- Targets on TI C6000 (DM6437), C2000



自动代码可靠性、效率

- “Our team at Honeywell Flight Controls has generated 1.6 million lines of code automatically using MathWorks products for the Joint Strike Fighter.”
- “We manually verified 1.2 million lines of code so far.”
- “We found 1 line of code that we are not sure where it came from or what it’s purpose is. Everything works fine, though.”
- “These days everyone is talking about doing 6-sigma design. One questionable line of code out of 1.2 million lines of code – that’s better than 6.5 sigma. And we are not talking about it – we actually did it.”

Honeywell

LOCKHEED MARTIN
We never forget who we're working for™

产品级代码

Pilot Result - Metrics

	Hand Code	Auto-code	Percent Change
Calibration ROM	9464	9464	0%
Code ROM	2952	2900	-1.76%
RAM	240	238	-0.83%

GM

MathworksMIAC21June05.ppt 8

SAE TECHNICAL PAPER SERIES

2004-01-0269

Multi-Target Modelling for Embedded Software Development for Automotive Applications

Grantley Hodge, Jian Ye and Walt Stuart

Table 2 shows ROM and RAM comparisons between hand code and auto code for a floating-point component in some typical powertrain software.

Table 2 ROM and RAM comparison between a floating-point hand code and auto code.

	Hand Code	Auto Code
ROM	6408	6192
RAM	132	112

The auto code has less size of ROM and RAM compared to that of hand code. The auto code is readable and peer reviewed, and checked with the QAC static analysis tool. Most importantly, the auto code is implemented in a real-world powertrain application.

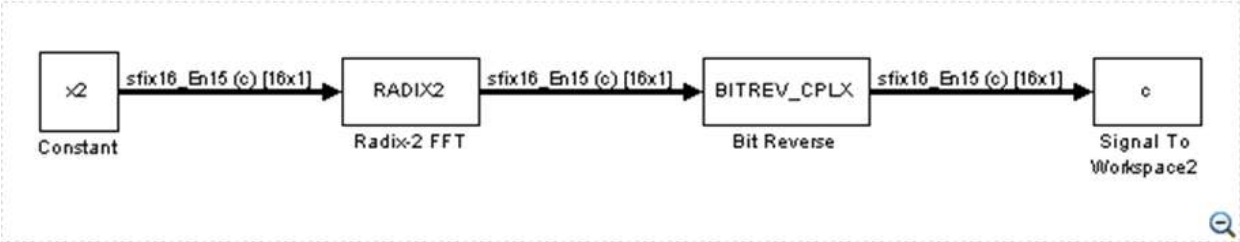
CONCLUSION

A custom data class allowing data type and data scaling

正确性和性能：

Examples

The Bit Reverse block reorders the output of the C64x Radix-2 FFT in the model below to natural order.



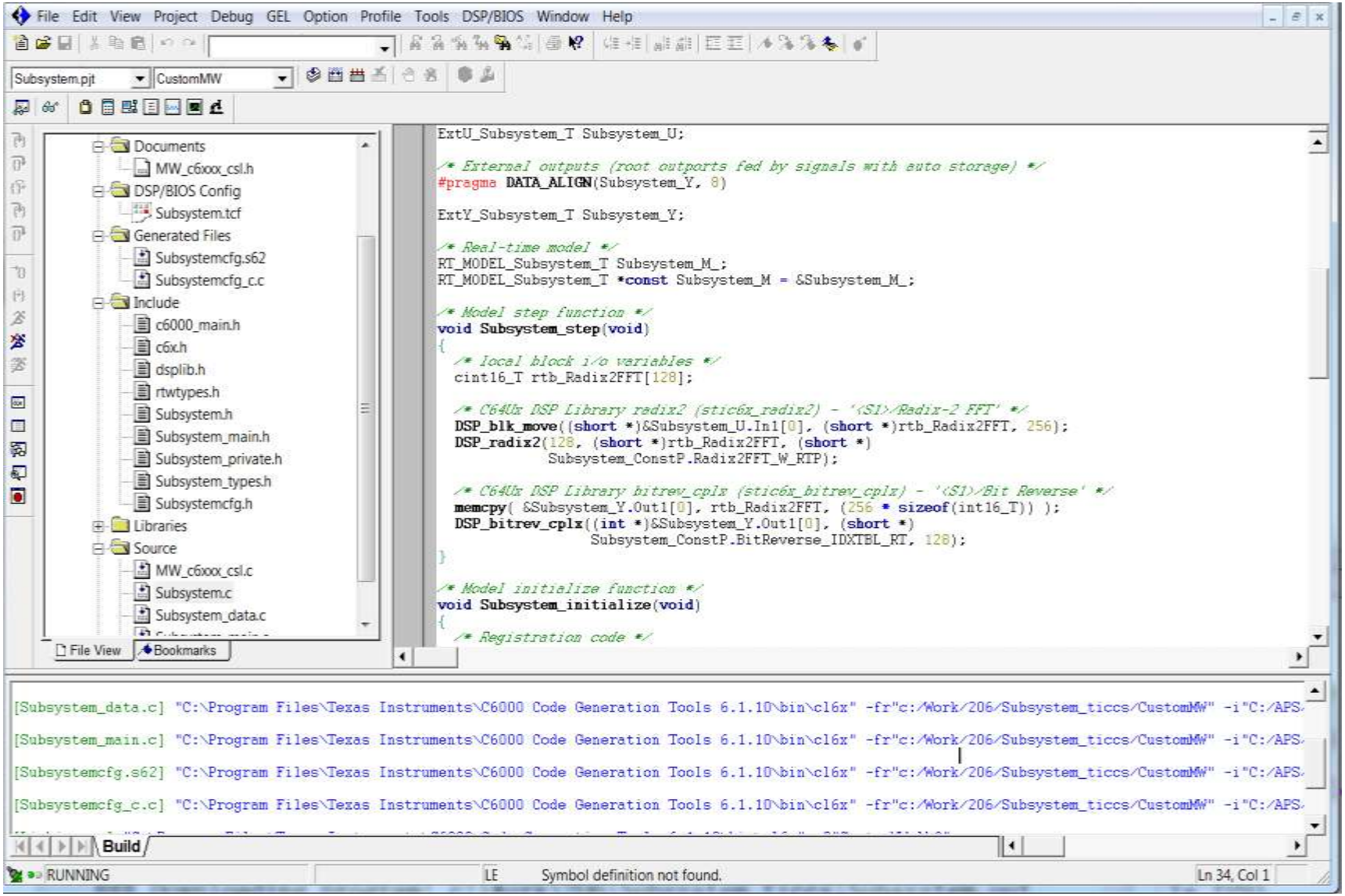
The following code calculates the same FFT in the workspace. The output from this calculation, y2, is displayed side-by-side with the output from the model, c. The outputs match, showing that the Bit Reverse block reorders the Radix-2 FFT output to natural order:

```
k = 4;
n = 2^k;
xr = zeros(n, 1);
xr(2) = 0.5;
xi = zeros(n, 1);
x2 = complex(xr, xi);
y2 = fft(x2);
```

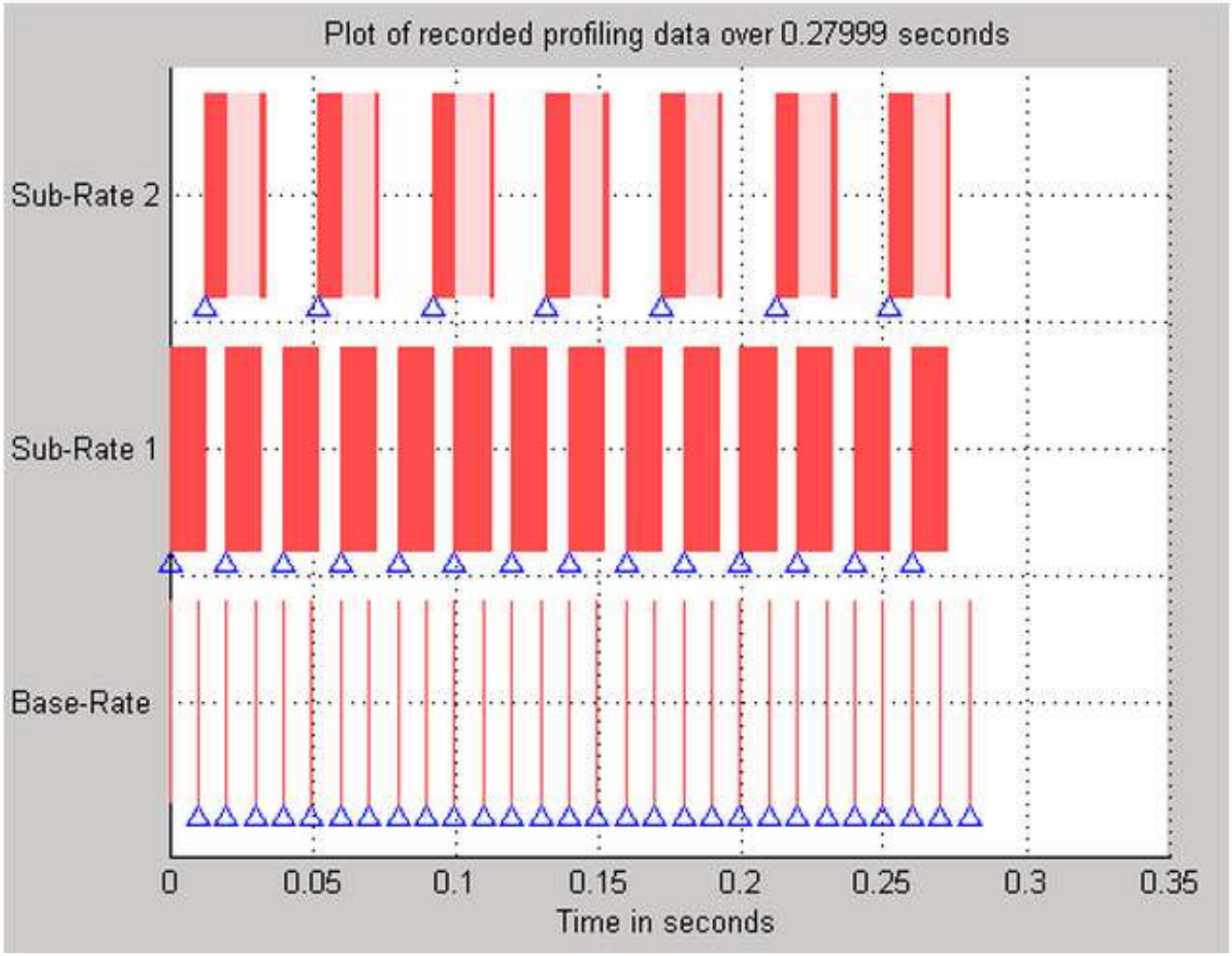
正确性：

[y2,	c]
0.5000	0.5000
0.4619 - 0.1913i	0.4619 - 0.1913i
0.3536 - 0.3536i	0.3535 - 0.3535i
0.1913 - 0.4619i	0.1913 - 0.4619i
0 - 0.5000i	0 - 0.5000i
-0.1913 - 0.4619i	-0.1913 - 0.4619i
-0.3536 - 0.3536i	-0.3535 - 0.3535i
-0.4619 - 0.1913i	-0.4619 - 0.1913i
-0.5000	-0.5000
-0.4619 + 0.1913i	-0.4619 + 0.1913i
-0.3536 + 0.3536i	-0.3535 + 0.3535i
-0.1913 + 0.4619i	-0.1913 + 0.4619i
0 + 0.5000i	0 + 0.5000i
0.1913 + 0.4619i	0.1913 + 0.4619i
0.3536 + 0.3536i	0.3535 + 0.3535i
0.4619 + 0.1913i	0.4619 + 0.1913i

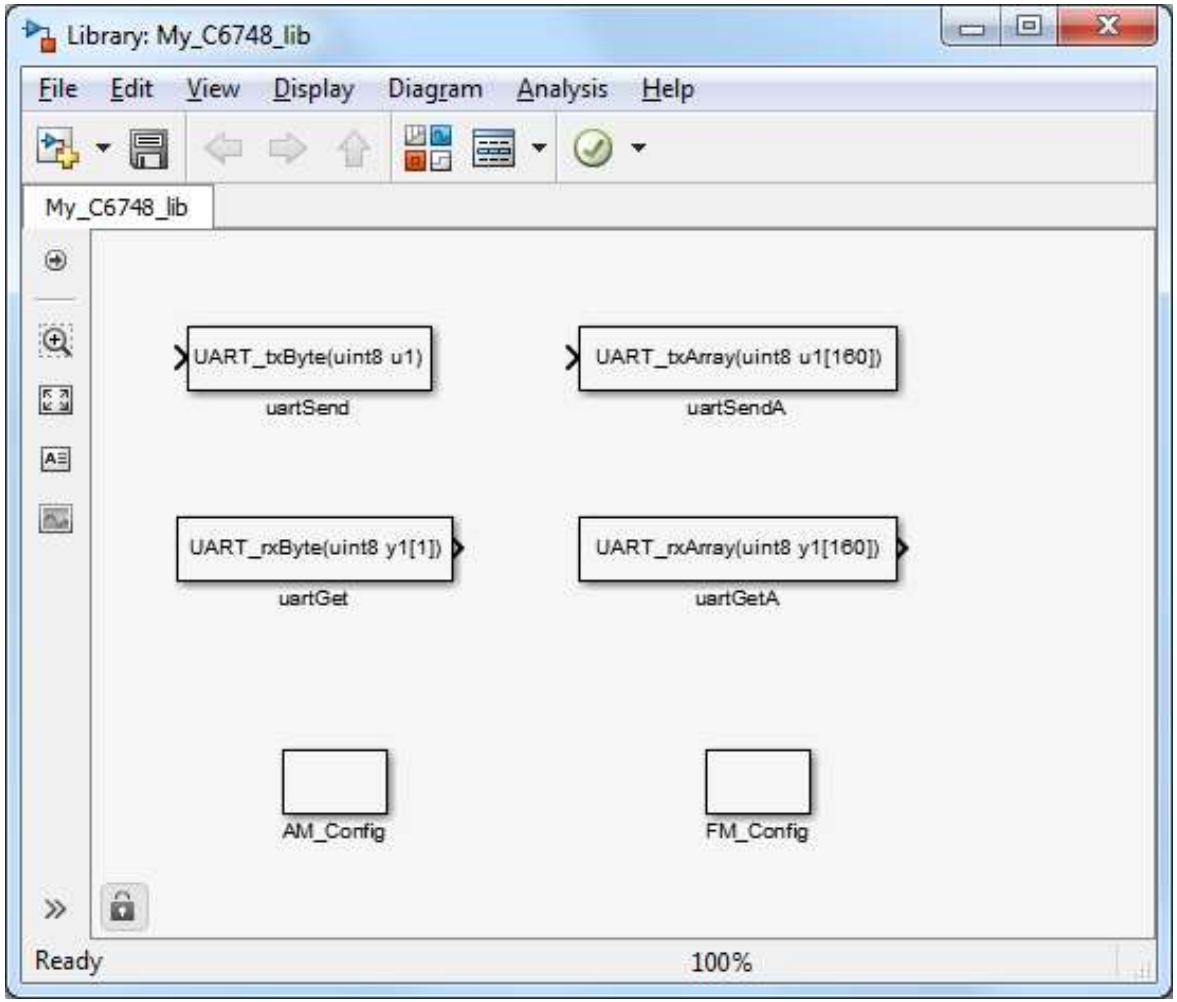
正确性和性能：代码


File Edit View Project Debug GEL Option Profile Tools DSP/BIOS Window Help
Subsystem.pjt
CustomMW
Documents
MW_c6xxx_csl.h
DSP/BIOS Config
Subsystem.tcf
Generated Files
Subsystemcfg.s62
Subsystemcfg_c.c
Include
c6000_main.h
c6x.h
dsplib.h
rtwtypes.h
Subsystem.h
Subsystem_main.h
Subsystem_private.h
Subsystem_types.h
Subsystemcfg.h
Libraries
Source
MW_c6xxx_csl.c
Subsystem.c
Subsystem_data.c
File View
Bookmarks
ExtU_Subsystem_T Subsystem_U;
/* External outputs (root outports fed by signals with auto storage) */
#pragma DATA_ALIGN(Subsystem_Y, 8)
ExtY_Subsystem_T Subsystem_Y;
/* Real-time model */
RT_MODEL_Subsystem_T Subsystem_M_;
RT_MODEL_Subsystem_T *const Subsystem_M = &Subsystem_M_;
/* Model step function */
void Subsystem_step(void)
{
/* local block i/o variables */
cint16_T rtb_Radix2FFT[128];
/* C64Dx DSP Library radix2 (stic6x_radix2) - '<S1>/Radix-2 FFT' */
DSP_blk_move((short *)&Subsystem_U.In1[0], (short *)rtb_Radix2FFT, 256);
DSP_radix2(128, (short *)rtb_Radix2FFT, (short *)
Subsystem_ConstP.Radix2FFT_W_RTP);
/* C64Dx DSP Library bitrev_cplx (stic6x_bitrev_cplx) - '<S1>/Bit Reverse' */
memcpy(&Subsystem_Y.Out1[0], rtb_Radix2FFT, (256 * sizeof(int16_T)));
DSP_bitrev_cplx((int *)&Subsystem_Y.Out1[0], (short *)
Subsystem_ConstP.BitReverse_IDXTBL_RT, 128);
}
/* Model initialize function */
void Subsystem_initialize(void)
{
/* Registration code */
[Subsystem_data.c] "C:\Program Files\Texas Instruments\C6000 Code Generation Tools 6.1.10\bin\cl6x" -fr"c:/Work/206/Subsystem_ticcs/CustomMW" -i"C:/APS
[Subsystem_main.c] "C:\Program Files\Texas Instruments\C6000 Code Generation Tools 6.1.10\bin\cl6x" -fr"c:/Work/206/Subsystem_ticcs/CustomMW" -i"C:/APS
[Subsystemcfg.s62] "C:\Program Files\Texas Instruments\C6000 Code Generation Tools 6.1.10\bin\cl6x" -fr"c:/Work/206/Subsystem_ticcs/CustomMW" -i"C:/APS
[Subsystemcfg_c.c] "C:\Program Files\Texas Instruments\C6000 Code Generation Tools 6.1.10\bin\cl6x" -fr"c:/Work/206/Subsystem_ticcs/CustomMW" -i"C:/APS
Build
RUNNING
LE
Symbol definition not found.
Ln 34, Col 1

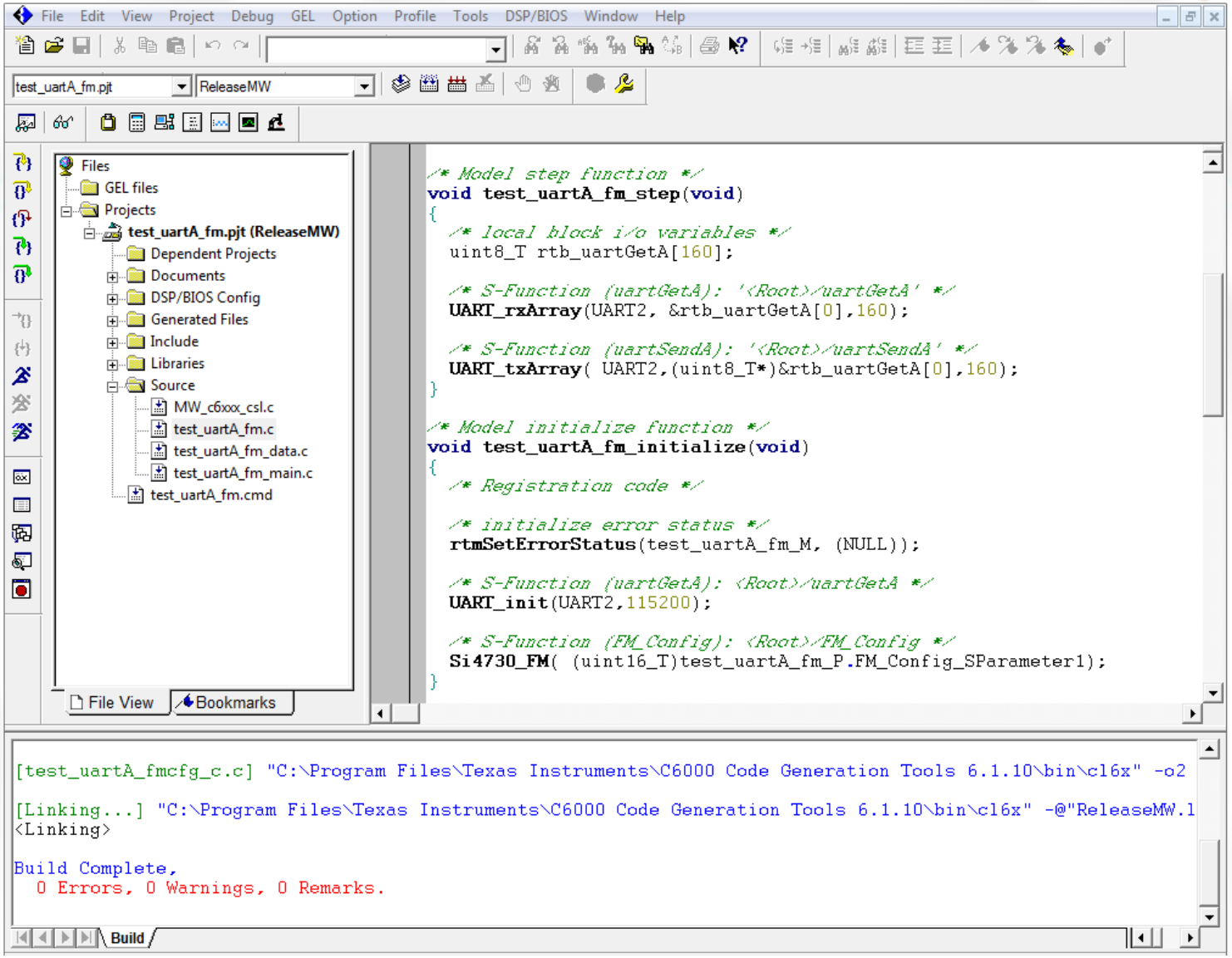
实时性：


Plot of recorded profiling data over 0.27999 seconds
Sub-Rate 2
Sub-Rate 1
Base-Rate
0
0.05
0.1
0.15
0.2
0.25
0.3
0.35
Time in seconds

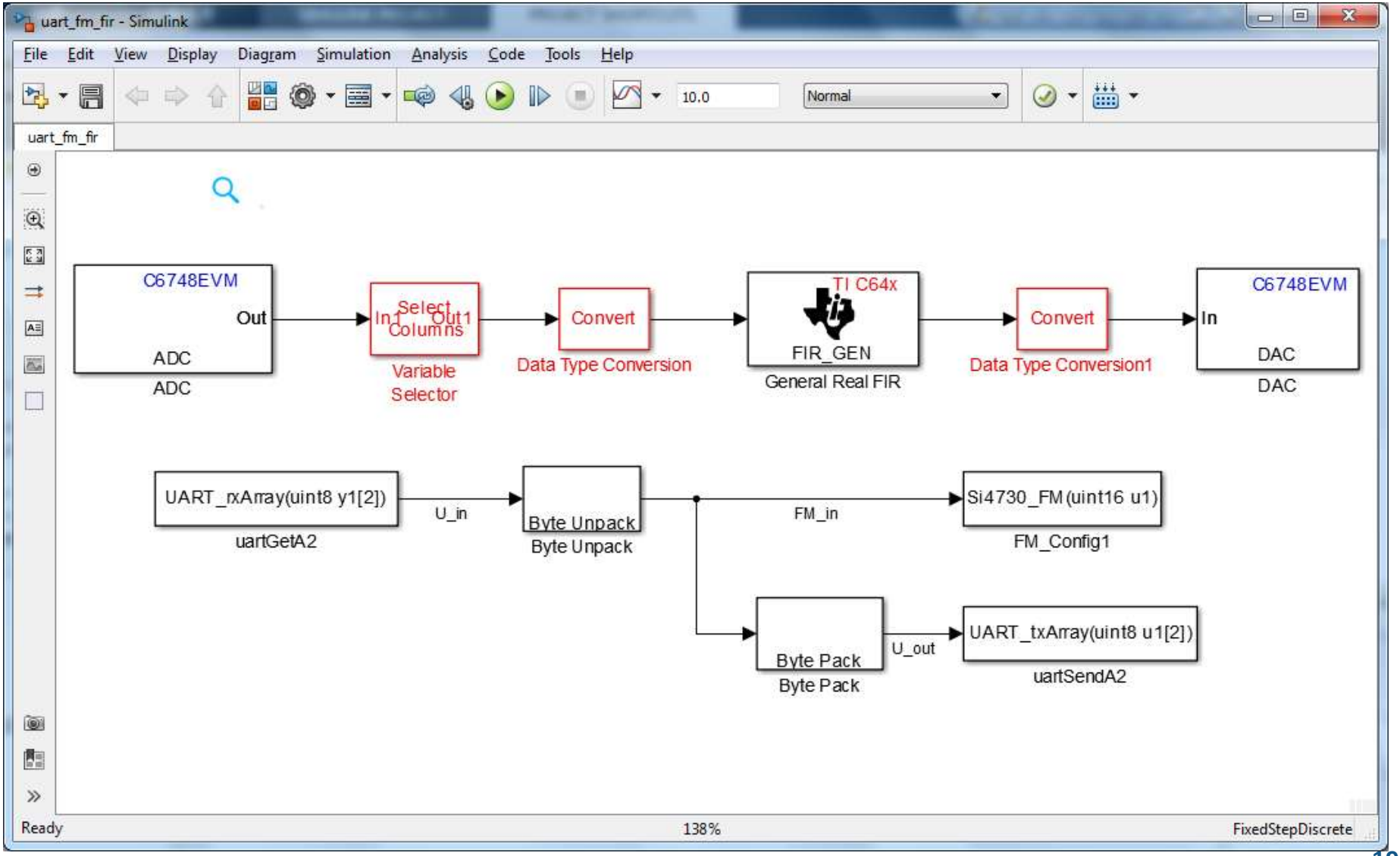
自定义模块：模块库


Library: My_C6748_lib
File Edit View Display Diagram Analysis Help
My_C6748_lib
UART_txByte(uint8 u1)
uartSend
UART_txArray(uint8 u1[160])
uartSendA
UART_rxByte(uint8 y1[1])
uartGet
UART_rxArray(uint8 y1[160])
uartGetA
AM_Config
FM_Config
Ready
100%

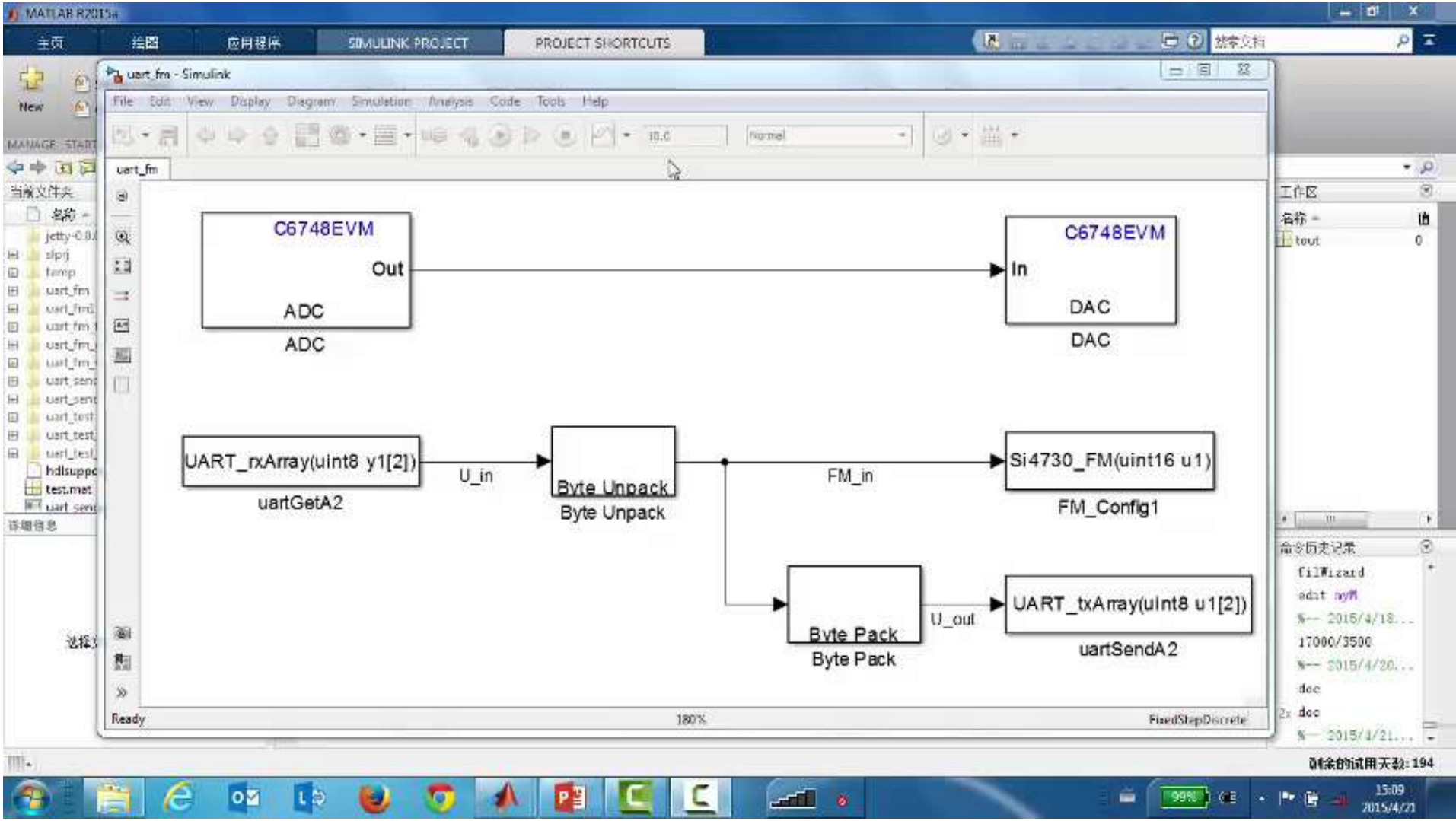
自定义模块：生成代码


File Edit View Project Debug GEL Option Profile Tools DSP/BIOS Window Help
test_uartA_fm.pjt
ReleaseMW
Files
GEL files
Projects
test_uartA_fm.pjt (ReleaseMW)
Dependent Projects
Documents
DSP/BIOS Config
Generated Files
Include
Libraries
Source
MW_c6xxx_csl.c
test_uartA_fm.c
test_uartA_fm_data.c
test_uartA_fm_main.c
test_uartA_fm.cmd
File View
Bookmarks
/* Model step function */
void test_uartA_fm_step(void)
{
/* local block i/o variables */
uint8_T rtb_uartGetA[160];
/* S-Function (uartGetA): '<Root>/uartGetA' */
UART_rxArray(UART2, &rtb_uartGetA[0],160);
/* S-Function (uartSendA): '<Root>/uartSendA' */
UART_txArray(UART2,(uint8_T*)&rtb_uartGetA[0],160);
}
/* Model initialize function */
void test_uartA_fm_initialize(void)
{
/* Registration code */
/* initialize error status */
rtmSetErrorStatus(test_uartA_fm_M, (NULL));
/* S-Function (uartGetA): <Root>/uartGetA */
UART_init(UART2,115200);
/* S-Function (FM_Config): <Root>/FM_Config */
Si4730_FM((uint16_T)test_uartA_fm_P.FM_Config_SParameter1);
}
[test_uartA_fmcfg_c.c] "C:\Program Files\Texas Instruments\C6000 Code Generation Tools 6.1.10\bin\cl6x" -o2
[Linking...] "C:\Program Files\Texas Instruments\C6000 Code Generation Tools 6.1.10\bin\cl6x" -@"ReleaseMW.l
<Linking>
Build Complete,
0 Errors, 0 Warnings, 0 Remarks.
Build

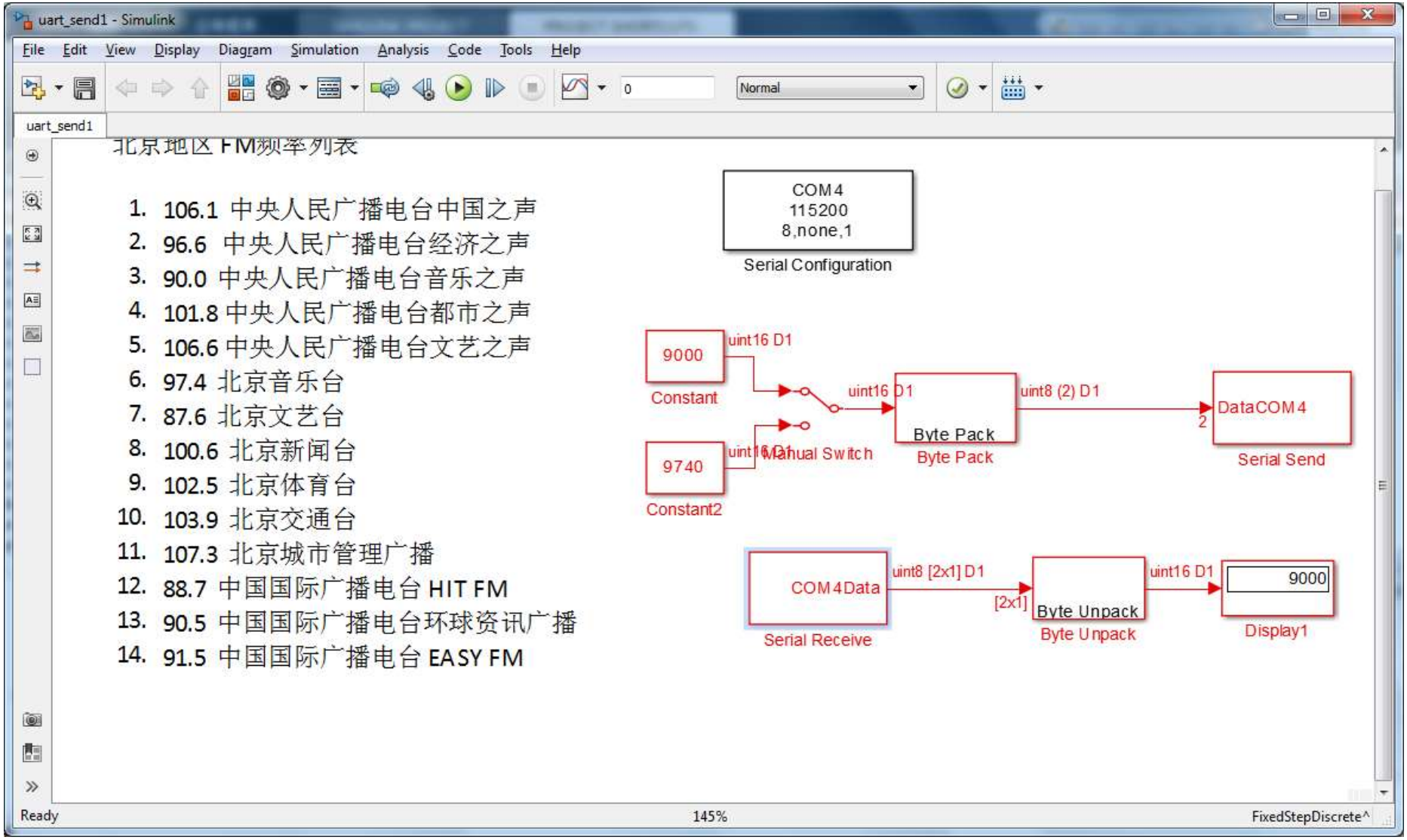
展台Demo：DSP 模型



代码产生的过程


uart_fm - Simulink
C6748EVM
Out
ADC
ADC
C6748EVM
In
DAC
DAC
UART_rxArray(uint8 y1[2])
uartGetA2
U_in
Byte Unpack
Byte Unpack
FM_in
Si4730_FM(uint16 u1)
FM_Config1
Byte Pack
Byte Pack
U_out
UART_txArray(uint8 u1[2])
uartSendA2
Ready
180%
FixedStepDiscrete

展台Demo：上位机


uart_send1 - Simulink
File Edit View Display Diagram Simulation Analysis Code Tools Help
Normal
uart_send1
北京地区FM频率列表
1. 106.1 中央人民广播电台中国之声
2. 96.6 中央人民广播电台经济之声
3. 90.0 中央人民广播电台音乐之声
4. 101.8 中央人民广播电台都市之声
5. 106.6 中央人民广播电台文艺之声
6. 97.4 北京音乐台
7. 87.6 北京文艺台
8. 100.6 北京新闻台
9. 102.5 北京体育台
10. 103.9 北京交通台
11. 107.3 北京城市管理广播
12. 88.7 中国国际广播电台 HIT FM
13. 90.5 中国国际广播电台环球资讯广播
14. 91.5 中国国际广播电台 EASY FM
COM4
115200
8,none,1
Serial Configuration
9000
Constant
uint16 D1
9740
Constant2
Manual Switch
Byte Pack
Byte Pack
uint8 (2) D1
DataCOM4
Serial Send
COM4Data
Serial Receive
uint8 [2x1] D1
[2x1]
Byte Unpack
Byte Unpack
uint16 D1
9000
Display1
Ready
145%
FixedStepDiscrete

展台：基于模型的设计在DSP的应用

谢 谢！