Model-Based Testing of Safety Critical Real-Time Control Logic Software

Yevgeny Gerlits, Alexey Khoroshilov
Institute for System Programming of RAS
AAFSS – Airborne Active Flight Safety System

1. AAFSS is developed by Russian Systems Corporation. The picture belongs to Russian Systems Corporation. The picture was taken from [http://www.rusys.ru](http://www.rusys.ru)
Control Logic Software (CLS) is a subsystem

Memory pool:

\[\text{sys\_time} \]

\[i_1, \ldots, i_n\]

\[s_1, \ldots, s_m\]

\[o_1, \ldots, o_k\]

TACT - Tact period
CLS is a number of decision control algorithms

We consider the following conditions:

- Boolean formulas;
- \( (\text{formula}(i_1,..,i_n), T) = \text{true} \) if Boolean formula has been true for \( T \) or more.

- go to algorithm \( j \);
- set values of output parameters or state variables;
- if condition then... else...
Real Time in CLS

- total time of all subsystems execution < tact period;
- temporal condition (formula($i_1,..,i_n$, $T$) = true if Boolean formula has been true for $T$ or more:

Example: ($i_1 < 5$, $2 \times TACT$)

Turns of the global control loop:
1. sys_time = 0; $i_1 = 3$;
2. sys_time = TACT; $i_1 = 2$;
3. sys_time = $2 \times TACT$; $i_1 = 3$;
4. sys_time = $3 \times TACT$; $i_1 = 6$. 
Temporal condition is closer to state than to real time

How does CLS calculate \((\text{formula}(i_1,\ldots,i_n), T)\)?

- Let \(\text{sys\_time}_f\) be \(\text{sys\_time}\) since when \(\text{formula}(i_1,\ldots,i_n)\) has been TRUE;
- \((\text{formula}(i_1,\ldots,i_n), T) = \text{formula}(i_1,\ldots,i_n) \&\& (\text{sys\_time} - \text{sys\_time}_f) \geq T\).
Characteristics of CLS

- ~ 2000 lines of code;
- Low Level Requirements are 9 flow charts of size A4;
- 32 input parameters of different types;
- 7 state variables of different types;
- 80 temporal conditions in branch instructions;
- 9 output parameters;
- tact period is 60 ms.
Problem Definition

- huge number of input parameters (32);
- huge space of states (7 state variables + 80 temporal conditions);
- CLS is a safety critical software (MC/DC metric)
  ⇒ Traditional unit testing doesn’t work well.

- real time characteristics of CLS are not complicated
  ⇒ Real Time specific MBT approaches (UPAAL Tron, Timed TorX) are not ultimately required.

- industrial tool is required in a real project
  ⇒ try a general purpose MBT (SpecExplorer, UniTESK).
On the Fly MBT Approach

\[ s' = \text{genState} (s_1, \ldots, s_m, \]
\[ (\text{formula}_1(i_1, \ldots, i_n), T_1), \ldots, \]
\[ (\text{formula}_p(i_1, \ldots, i_n), T_p)) \]

MBT test system
Oracle

\[ (\text{formula}_{i1}(i_1, \ldots, i_n), T_{i1}), \ldots, (\text{formula}_{p0}(i_1, \ldots, i_n), T_{p0}) \]

\[ \text{CLS under test} \]

\[ (\text{formula}_{i11}(i_1, \ldots, i_{n1}), T_{i11}), \ldots, (\text{formula}_{p1}(i_1, \ldots, i_{n1}), T_{p1}) \]

\[ \{\text{true, false}\} \]
MBT Scheme

subsystem 1

i_{11}, \ldots, i_{n_1}

CLS

sys\_time, o_{11}, \ldots, o_{k_1}, s_{11}, \ldots, s_{m_1}

TACT

subsystem 2

i_{11}, \ldots, i_{n_1}

Test adaptor

(f_{11}, T_{11}), \ldots, (f_{p_1}, T_{p_1})

Oracle

(f_{11}, T_{11}), \ldots, (f_{p_1}, T_{p_1})

s_{11}, \ldots, s_{m_1}

Generator

s_0', s_1', s_2', i_{12}, \ldots, i_{n_2}

UniTESK MBT Scheme

Target computer system (device) Host computer system (Server)

physical channel

subsystem 1
Set-mediator
CLS
Get-mediated subsystem 2

Mediator
Oracle
Test sequence iterator
Test engine
Conclusion

1. The RTCLS subsystem and the architecture of the whole embedded device were described;
2. An MBT approach to RTCLS was outlined in general terms;
3. The MBT approach was implemented using UniTESK.

⇒ general purpose MBT like UniTESK are applicable to CLS.
Thank you! Questions?