Verifysoft TECHNOLOGY

НПП ИТЭЛМА

Testwell CTC++ User Testimonial

1. Figure - Itelma building

NPP Itelma is the leading developer and manufacturer of serial automotive electronics in Russia. Founded in 1994, the company has today 3 production sites in Moscow, Dimitrovgrad and Kostroma, as well as the 4 R&D centers. ITELMA supplies components to all big Russian car manufacturers like Lada, Kamaz, Ulyanovsky Avtomobilny Zavod (UAZ), Gorkowski Awtomobilny Sawod (GAZ), and Pawlowski Awtobusny Sawod (PAZ).

The company delivers also special equipment for the satellite navigation system GLONASS and is as the general partner to a number of truck industry‘ players in development of advanced electronic architecture.

ГАЗ ПАЗ УАЗ KAMAZ LADA

2. Figure - Customers of Itelma



3. Figure - Itelma work station

NPP Itelma has modern laboratories with all the necessary equipment for carrying out the full cycle of testing of the automotive electronic components.

Our production has most modern equipment that allows us to produce high quality electronic components in multi-million batches.

Verifysoft
TECHNOLOGY

Safety and reliability of each electronic unit is very important in automotive field. Just imagine what will happen if, for example, an uncontrolled acceleration of a car happens, or, for example, at the moment of overtaking in the oncoming lane, airbags will deploy or the windscreen wipers will fail during the heavy rain.

Every year the complexity and functionality of the cars are increasing, and as a result, the total amount of embedded software in the cars is getting more. Accordingly, the risk of injury and even death due to software errors rises by many times. Errors in the software can be the cause of the recall of millions of cars, as a result of which the development company may suffer huge losses, up to complete bankruptcy. To confirm the importance of reliability and safety of automotive software, here are some statistics of car reviews:

- 2001, Ford, recall of 22 million cars, ignition system defect
- 2009, Ford, recall of 15 million cars, a defect in the cruise control system, a short circuit in which caused a fire
- 2009, Toyota, recall of 9 million cars, uncontrollable car acceleration
- 2013, АвтоВАЗ, recall of 30 thousands cars, problems with the brake system
- 2014, General Motors, recall of 2,4 million cars, transmission defect
- 2015, Takata, recall of 34 million cars due to airbag defects (Honda, BMW, Fiat, Mazda, Mitsubishi, Nissan, Subaru, Toyota, Chrysler, General Motors

A modern car contains dozens of electronic components and hundreds of millions of lines of code (for comparison, 2–3 million lines of code in modern aircraft), and every line of code must be thoroughly tested. In the process of software development, NPP Itelma adhere to the V-cycle, which is the de facto standard in the automotive industry. One of the most important stages of the V-cycle is unit testing of software. Our department is developing embedded software. Each software module in the development process is covered by unit tests that are developed based on requirements. First, we test the software module on the black box principle to confirm its correct functioning. But this is not enough. For deeper and more complete testing, we test the code on the white box principle. And here, without a tool analyzing code coverage with tests, it is difficult or even impossible to assess how fully the unit testing has been completed. To analyze the code coverage, we tried many different tools and selected finally Testwell CTC++. It is very simple and easy to use, and at the same time very fast and efficient. Testwell CTC++ generates very simple, clear and easy-to-analyze coverage reports. In addition, this tool can be used to certify safety-critical projects, which is an important criterion for choosing a tool for automotive applications. With the help of Testwell CTC++, we found many uncovered sections of code and conditions.

Instrumentation mode: multicondition
TER: 92 % (174/190) structural, 93 % (275/296) statement
To files: Previous | Next

TER % - multicondition		TER % - statement		Calls	Line	Function
100 %	(4/4)	100 %	(7/7)	144	239	FLASH_Init()
100 %	(4/4)	100 %	(4/4)	144	288	FLASH_DeInit()
100 %	(14/14)	100 %	(20/20)	52	331	FLASH_SetJobCallback()
90 %	(18/20)	100 %	(23/23)	48	404	FLASH_Read()
100 %	(14/14)	100 %	(20/20)	48	524	FLASH_Write()
100 %	(6/6)	100 %	(8/8)	16	629	FLASH_GetJobResult()
100 %	(4/4)	100 %	(5/5)	12	680	FLASH_GetMemoryStatus()
100 %	(4/4)	100 %	(3/3)	8	720	FLASH_SetMemoryStatus()
100 %	(12/12)	100 %	(19/19)	48	760	FLASH_IsAreaBlank()
100 %	(4/4)	100 %	(6/6)	8	852	FLASH_GetSectorSize()
100 %	(15/15)	100 %	(20/20)	44	894	FLASH_Erase()

4. Figure - Testwell CTC++ report (summary)

```
                 404 STD_RESULT FLASH_Read(const U32 nSourceAddress,
                 405                       U8* const pDataBuffer,
                 406                       const U32 nDataQty)
                 407 {
                 408     STD_RESULT nFuncResult = RESULT_OK;
                 409     #if (ON == FLASH_DEVELOPMENT_ERROR_DETECTION)
      44     4   410     if (TRUE == FLASH_bInitialized)
                 411     {
      40     4   412         if (0U != nDataQty)
                 413         {
                 414     #endif //(ON == FLASH_DEVELOPMENT_ERROR_DETECTION)
      32     8   415             if (NULL_PTR != pDataBuffer)
                 416             {
      28     4   417                 if (FLASH_JOB_IDLE == FLASH_nCurrentJob)
                 418                 {
                 419                     if ((TRUE == FLASH_RangesAreCorrect(nSourceAddress,
                 420                                                         (nSourceAddress + nDataQty) - 1U)) ||
                 421                         ((0U == nDataQty) && (TRUE == FLASH_RangesAreCorrect(nSourceAddress,
      16    12   422                                                              (nSourceAddress + 1U)))))
      16         422             1: (T) || ((_) && (_))
       0         422             2: (F) || ((T) && (T))
           0     422             3: (F) || ((T) && (F))
          12     422             4: (F) || ((F) && (_))
                 423                 {
                 424                     //Initialize internal parameters for HW-level driver.
                 425                     FLASH_SetInitialParameters(nSourceAddress,
                 426                                                nDataQty,
                 427                                                pDataBuffer);
```

5. Figure - Testwell CTC++ report (source code view)

The missing tests were written in time, which allowed to detect many errors in the early stages of development. As a result, the overall quality of the firmware has improved significantly. The Testwell CTC++ tool thus took its honored place in our V-development cycle.

Alexander Dolozov,
Automotive Embedded Software Manager,
NPP ITELMA LLC
Russia

C Photos: Itelma and Verifysoft Technology GmbH