



Интегрированная среда разработки CM-LYNX

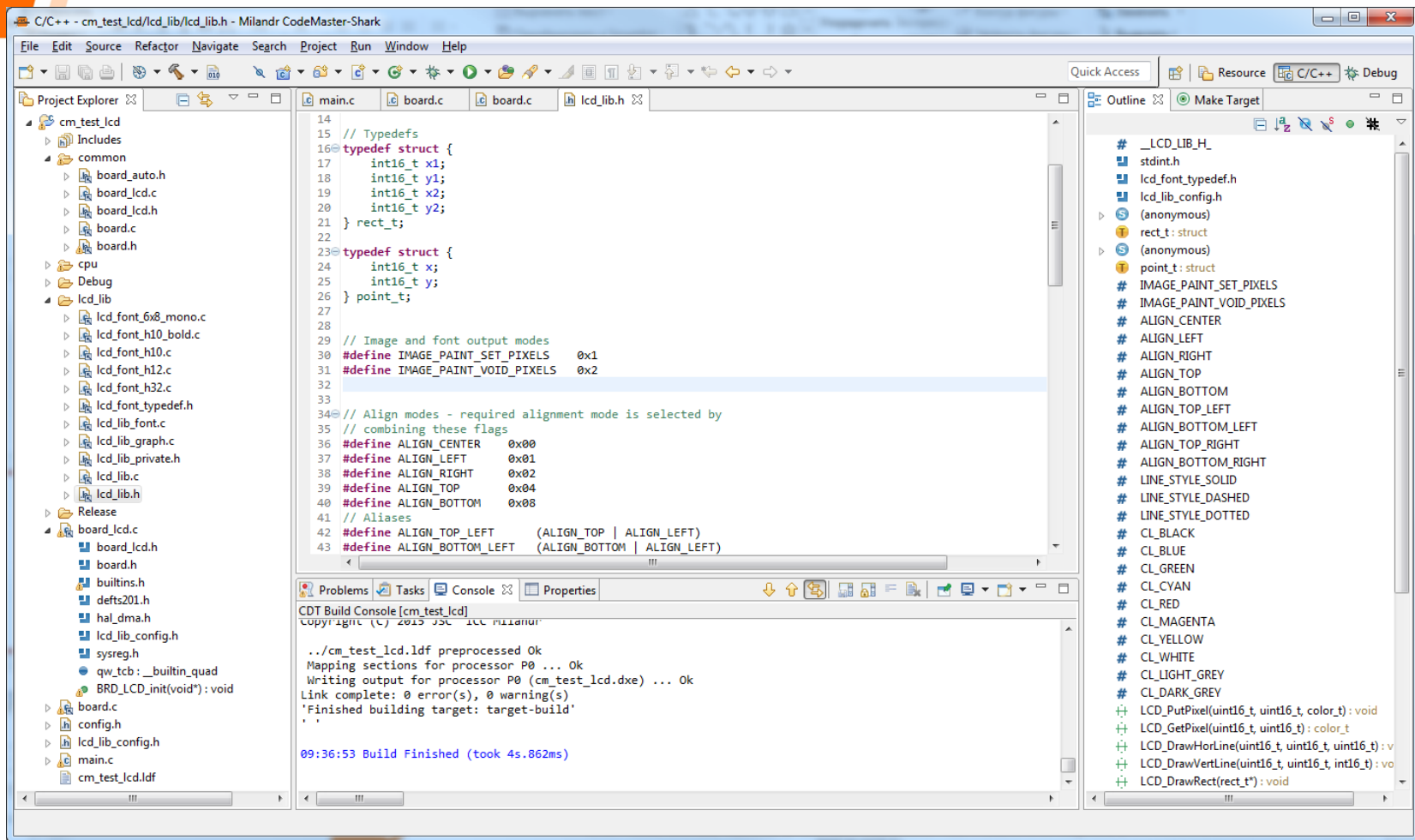
Афанасьев Станислав Владимирович

Казань, 11 августа 2016 г.

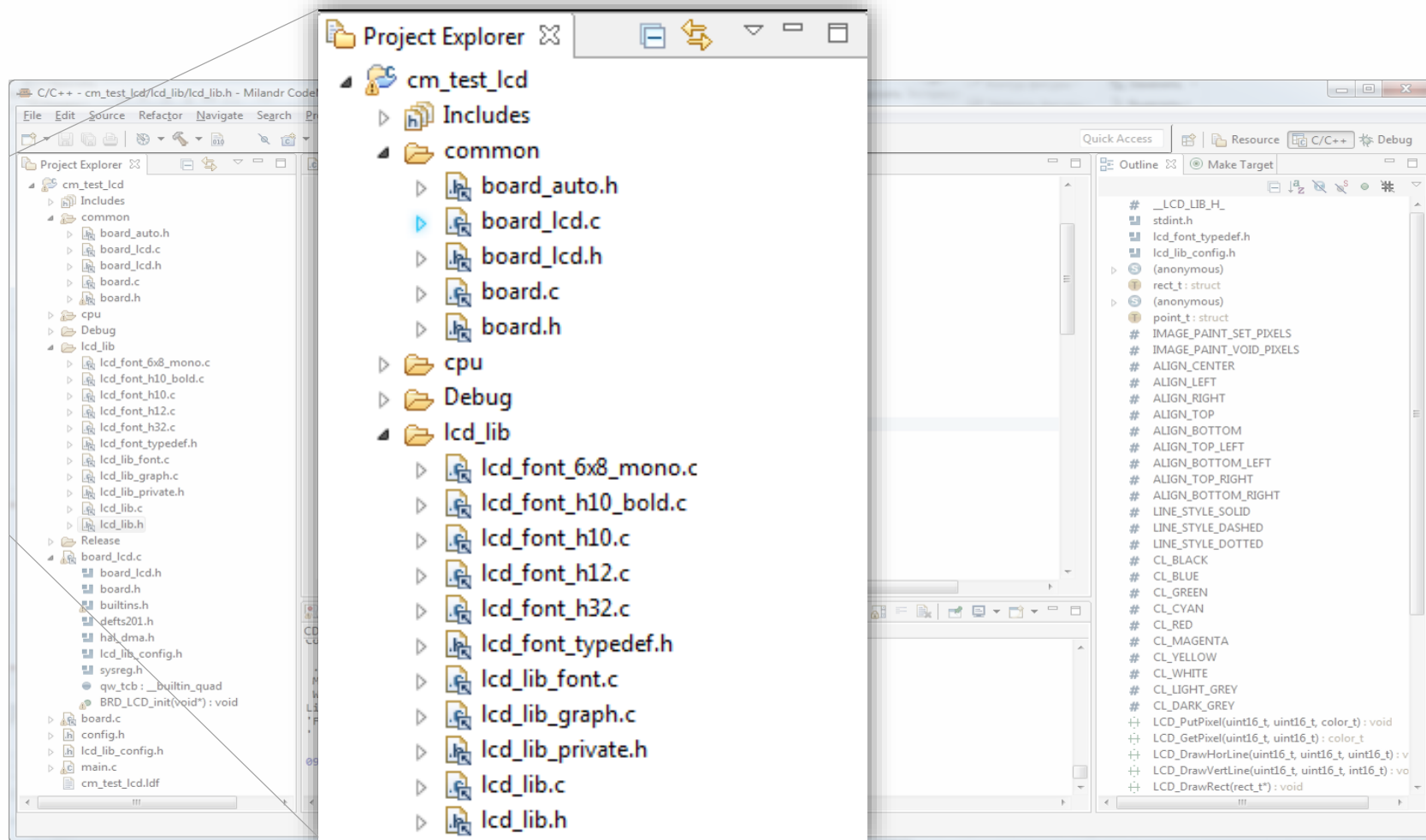
Полный цикл разработки ПО для процессоров ЦОС 1967ВЦ2, 1967ВЦ3

- Кроссплатформенная среда — поддержка ОС Windows, Linux
- Ассемблер, компилятор C/C++
- Полностью свой toolchain с поддержкой расширенного набора инструкций
- Платформа Eclipse
- Программный симулятор
- Аппаратный отладчик
- Поддержка многопроцессорной отладки
- Связь с Matlab для PIL симуляции и отладки алгоритмов
- Утилиты для формирования загрузочного образа и прошивки ПЗУ
- Набор программных библиотек

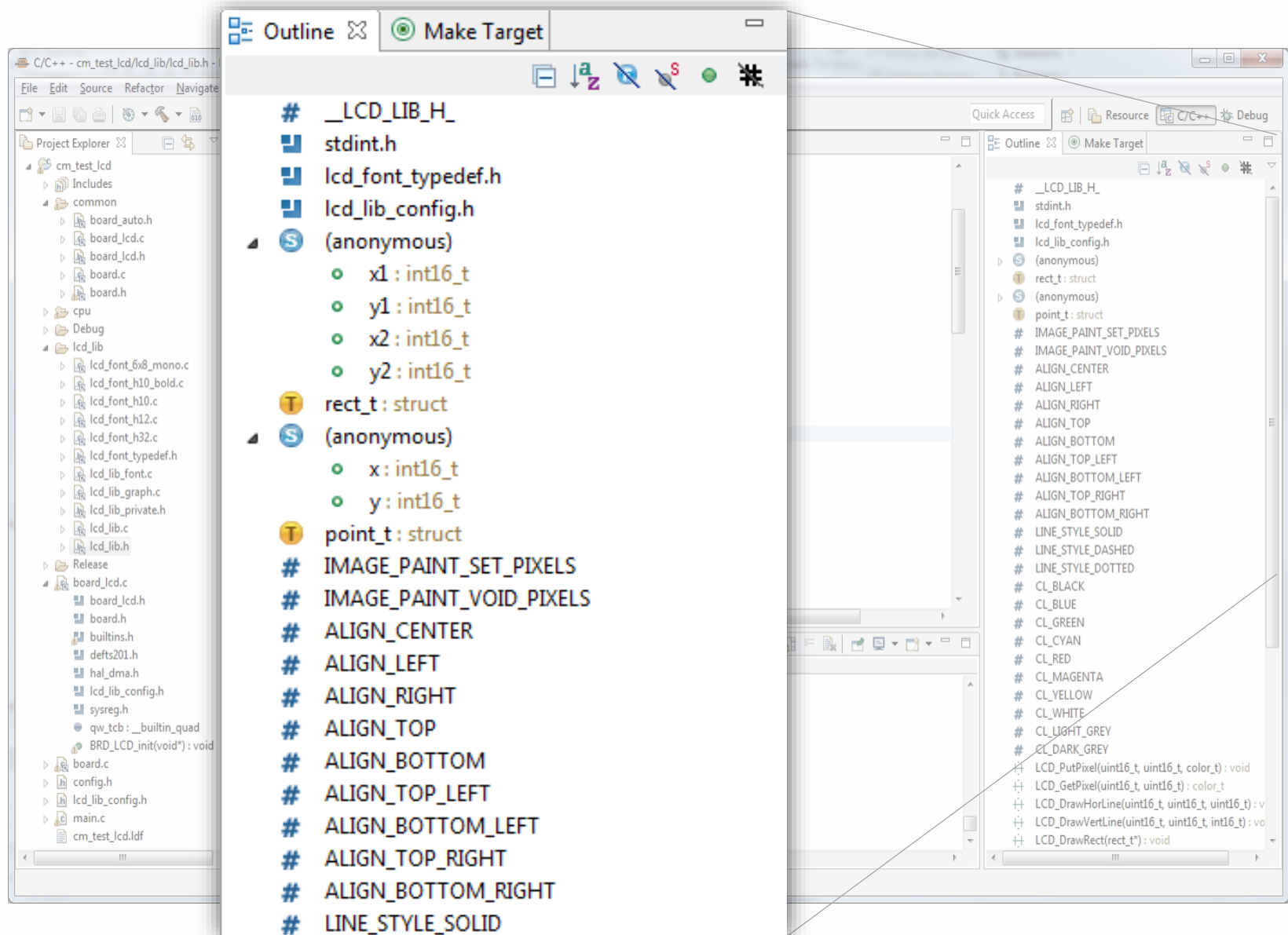
Общий вид среды CM-LYNX



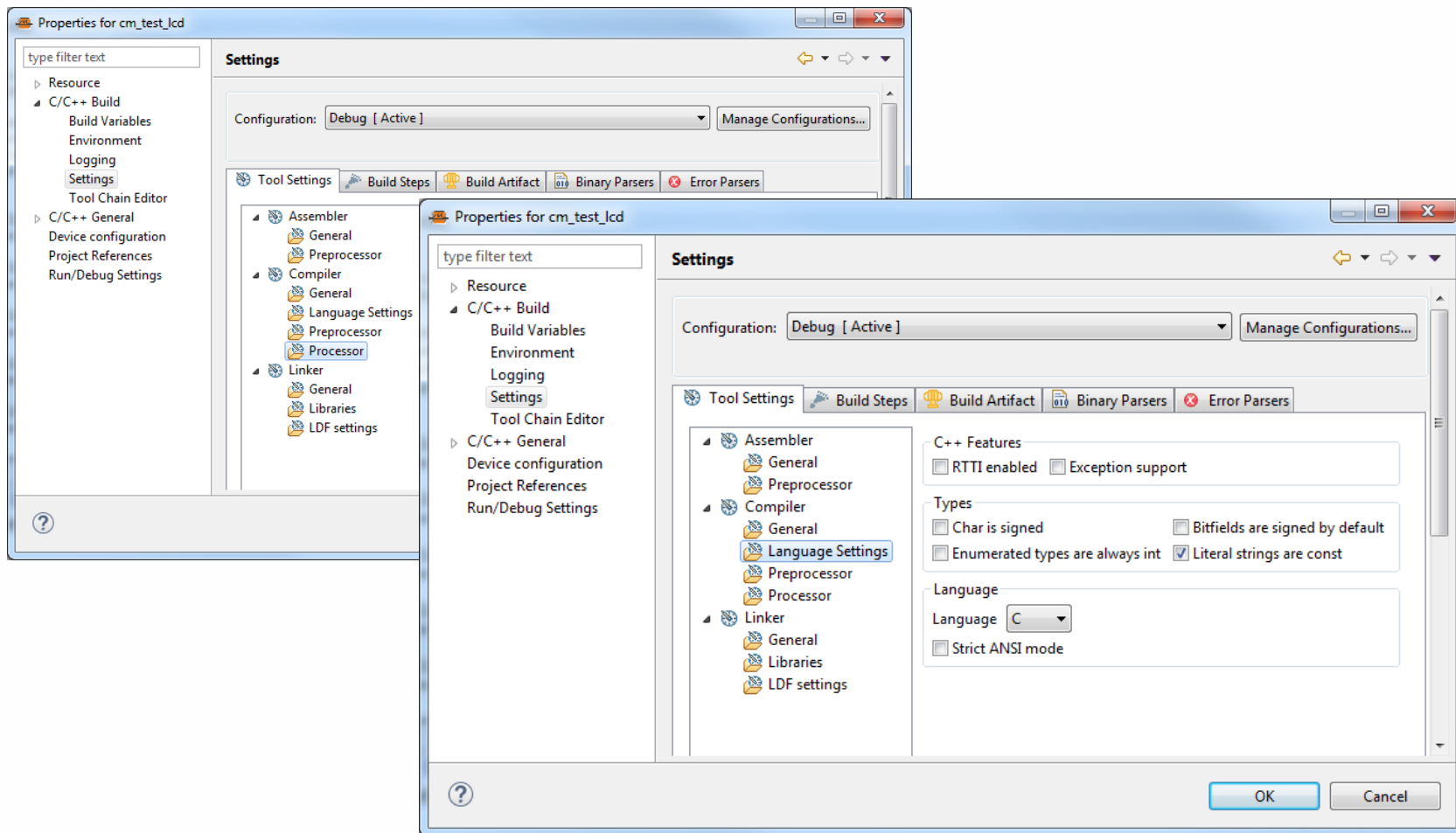
Дерево проектов и файлов



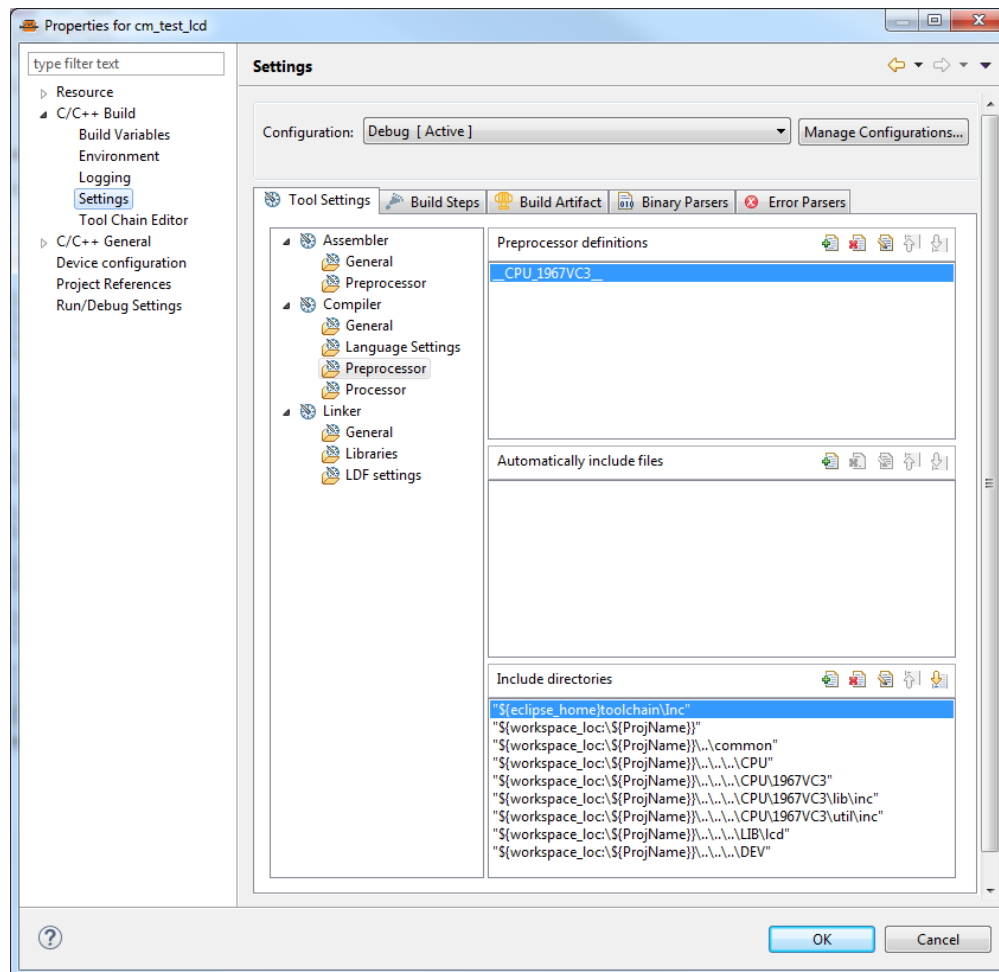
Выделение синтаксических элементов



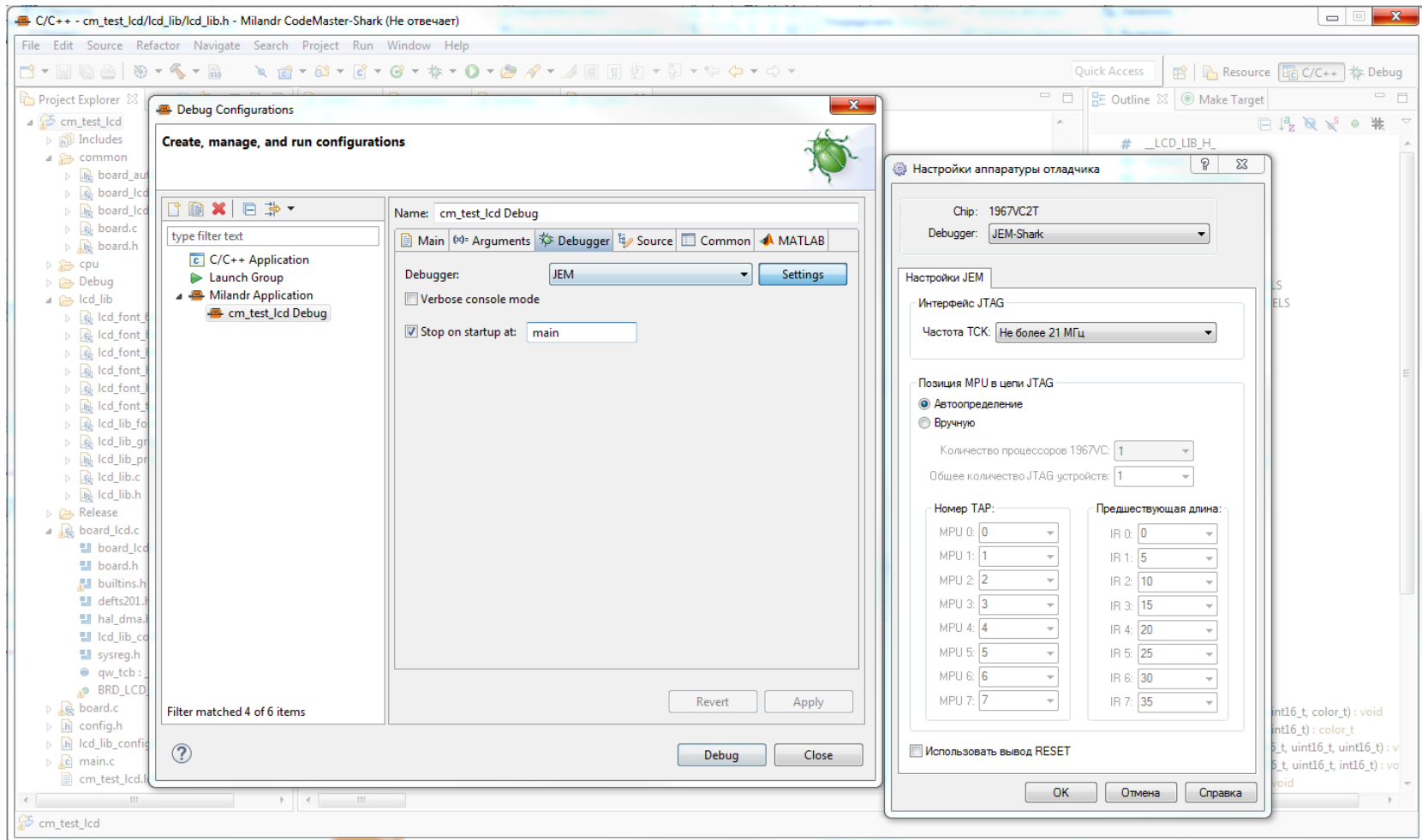
Настройки компилятора



Определение путей к заголовочным и исходным файлам



Настройка отладочной сессии



Отладка в симуляторе на аппаратной платформе

Debug - cm_test_lcd/board.c - Milandr CodeMaster-Shark

File Edit Source Refactor Navigate Search Project Run Window Help

Quick Access Resource C/C++ Debug

Debug main.c board.c board.c

cm_test_lcd Debug [Milandr Ap
gdb/mi (11.03.16, 9:12) (Sus
Thread [0] (Suspended)
1 <symbol is not avz
C:\CM-SHARK\GDB_ML.exe
E:\REA_SOFT\LynxToolbox\

```
75 //PX_ALT_PDB23F0 | // DATA[31:16] =  
76 0; // DATA[31:16] =  
77  
78 // Enable global interrupts  
79 __builtin_sysreg_write(_SQCTLST, SQCTL_GIE);  
80 }  
81  
82 void BRD_PLL_init(uint32_t f_ref, uint32_t f_core, uint32_t f_bus)  
83 {  
84 HAL_PLL_ConfigCalc(&pll_conf, f_ref, f_core, 1);  
85 LX_CMU->cpuPll.reg = pll_conf.value;  
86  
87 HAL_PLL_ConfigCalc(&pll_conf, f_ref, f_bus, 1);  
88 LX_CMU->busPll.reg = pll_conf.value;  
89  
90 LX_CMU->clockCfg.reg = (CPU_CPLL_SEL | CPU_BPPLL_SEL | CPU_BCLK_SEL_BPPLL);  
91  
92 HAL_PLL_pause(100);  
93 }  
94  
95  
96 void BRD_Link_PLL_Init(uint32_t f_ref, uint32_t f_link)  
97 {  
98 HAL_PLL_ConfigCalc(&pll_conf, f_ref, f_link, 1);  
99 LX_CMU->linkPll.reg = pll_conf.value;  
100 }  
101  
102  
103 void BRD_SDRAM_init(void)  
104 {  
105 uint32_t temp32u;  
106  
107 //-----//
```

00001672: J0 = J0 AND -5 (NF);
69 uart->brate.reg = (xref * 1000) / baudrate - 1;
00001675: XR0 = 0x7fff00;;
73 }
00001679: XR0 = 0x7f;;
79 // Disable UART interface
0000167c: XR0 = 4;;
80 uart->cr.reg = 0;
0000167e: J27 = J27 + 4 (NF);
83 {
BRD_PLL_init:
00001680: J27 = J27 - 4 (NF);
00001682: [J27 + 1] = XR25;;
00001684: XR24 = J4;;
00001686: J6 = J5;;
84 HAL_PLL_ConfigCalc(&pll_conf, f_ref, f_core, 1);
00001689: J5 = XR24;;
0000168b: CALL <HAL_PLL_ConfigCalc>;
85 LX_CMU->cpuPll.reg = pll_conf.value;
0000168f: [J31 + 0x800001c4] = XR0;;
87 HAL_PLL_ConfigCalc(&pll_conf, f_ref, f_bus, 1);
00001693: J5 = XR24;;
00001695: J7 = 1;;
88 LX_CMU->busPll.reg = pll_conf.value;
00001698: XR0 = [J31 + pll_conf + 0x2];
90 LX_CMU->clockCfg.reg = (CPU_CPLL_SEL | CPU_BPPLL_SEL);
0000169c: XR0 = 3;;
92 HAL_PLL_pause(100);
0000169f: J4 = 0x64;;
93 }
000016a2: XR24 = [J27 + 0];

Console Tasks Problems Executables Memory

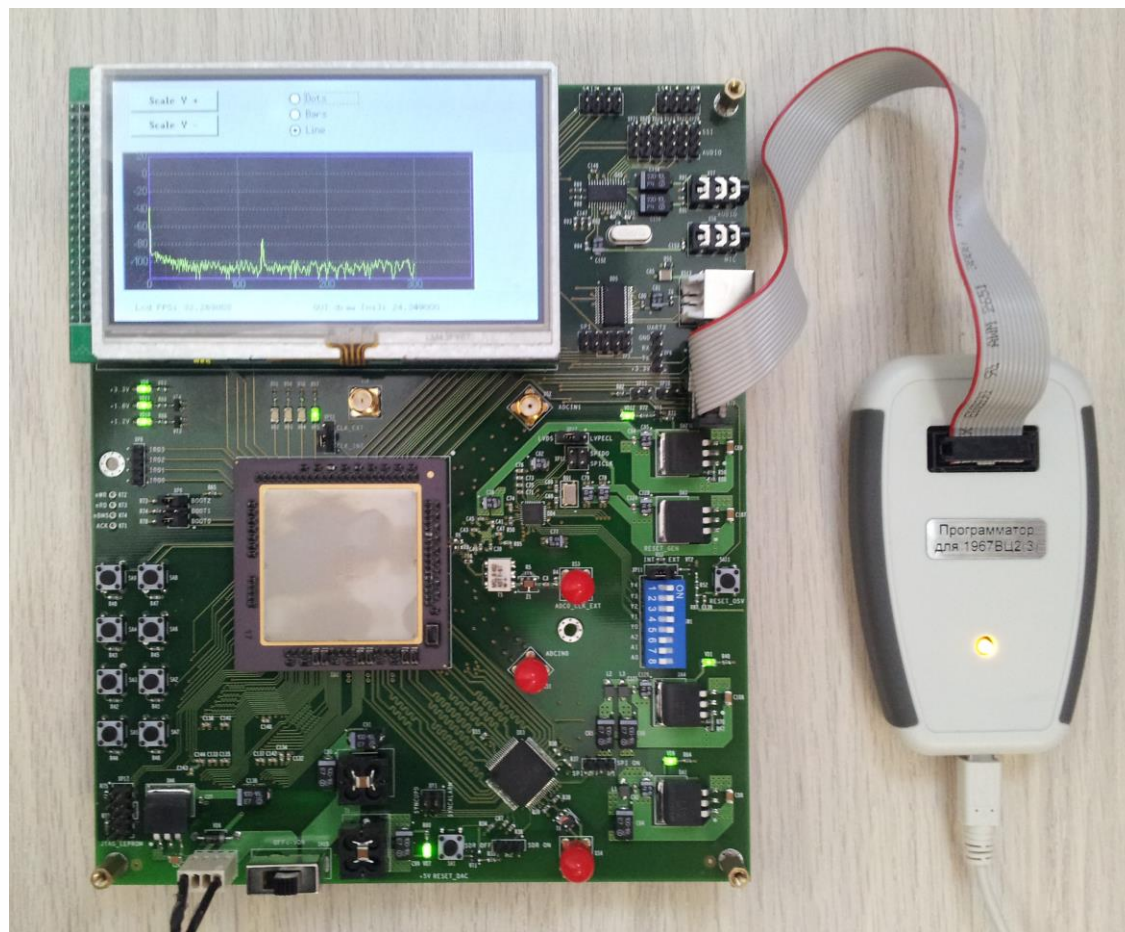
Monitors

0x00001700

Address	0	1	2	3	4	5	6	7
00001700	F1000011	8D002403	71000013	33C00000	B3C00000	8D002401	71400011	33C00000
00001708	B3C00000	08008000	B2400000	7108000C	B3C00000	08008000	B2800000	71080008
00001710	B3C00000	08008000	B2100000	71080004	B3C00000	08008000	B2200000	04003F0C
00001718	B2800010	F3084000	33C00000	B3C00000	CC3B3B14	84183B12	87403B13	89840030
00001720	CC003B04	CFE04200	88008024	A1002000	0800D090	B2000300	84003B05	88009388
00001728	84003B06	8800800C	84003B07	81803B08	CE6020F7	85003B08	81803B08	CE402010

Writable Smart Insert 75:73

Отладка проекта через JTAG эмулятор



- Стандартные библиотеки C/C++
- Набор DSP библиотек (фильтрация, БПФ, работа с матрицами)
- Библиотеки StdPeriphLibrary (HAL) для аппаратных блоков:
 - DMA
 - LCD
 - UART
 - SPI
 - SSI
 - PLL
 - CMU
 - LINK
 - Up/Down Converter
- ОС PB MilandrOS

124498, г. Москва, Зеленоград,
Георгиевский пр-т, д. 5

Тел.: +7 (495) 981-54-33
Факс: +7 (495) 981-54-36

info@milandr.ru

WWW.MILANDR.RU