文档编号：AN2032

上海东软载波微电子有限公司

用户手册

ES-DAP-Viewer

修订历史

版本	修改日期	更改概要
V1.0	2021-08-28	初版发布

地　址：中国上海市龙漕路 299 号天华信息科技园 2A 楼 5 层
邮　编：200235
E-mail：support@essemi.com
电　话：+86-21-60910333
传　真：+86-21-60914991
网　址：http://www.essemi.com/

目 录

内容目录

图目录

表目录

第1章　简介

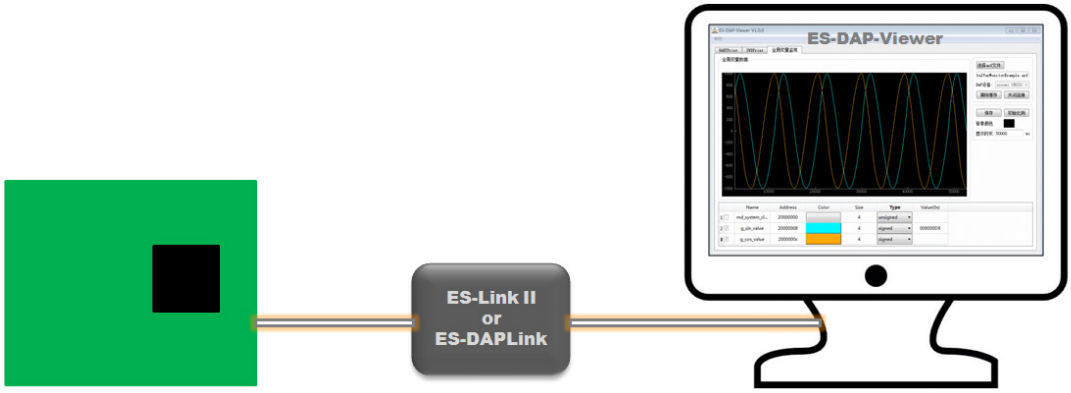

图 1　ES-DAP-Viewer 使用示意图

MCU 向调试终端输出信息的方法有很多。ES-DAP-Viewer 是一个更炫更酷、可以图形化显示数据的调试软件，它可以在目标 MCU 运行时，实时分析数据并图形化显示的 PC 端软件。用户可以简单的将目标 ES32 芯片连接到 ES-Link II 调试器或 ES-DAPLink 调试器，并启动 ES-DAP-Viewer 软件，就可以像示波器一样显示多个变量的值。它支持通过如下三种模式获取数据：

1. **UART-Print 模式**：用户通过串口将数据以固定的格式发送到 PC 端，上位机软件按照用户界面属性配置将收到的数据绘成波形。

2. **SWD-Print 模式**：这种模式不要额外的 UART 引脚， PC 软件通过 SWD 调试接口获取数据，用户只需要使用 essemi_swd_printf 函数将数据写入目标缓冲区即可。

3. **全局变量监视模式**：这种模式也是通过 SWD 调试接口获取数据，并通过读取一个 axf 文件，允许选择一定数量的变量可视化。

双击安装文件 ES-DAP-Viewer.，同意安装协议，选择安装路径，点击确定后等待软件安装，安装完成后即可在桌面看到 ES-DAP-Viewer 快捷方式。

图 2　ES-DAP-Viewer 启动图标

注 1：安装完成后，可以在安装路径的“sample_project”文件夹下找到 MCU 的固件程序用例；
注 2：“SWD-Print 模式”和“全局变量监视模式”无法和 IDE 调试同时使用。

第2章 UART-Print 模式

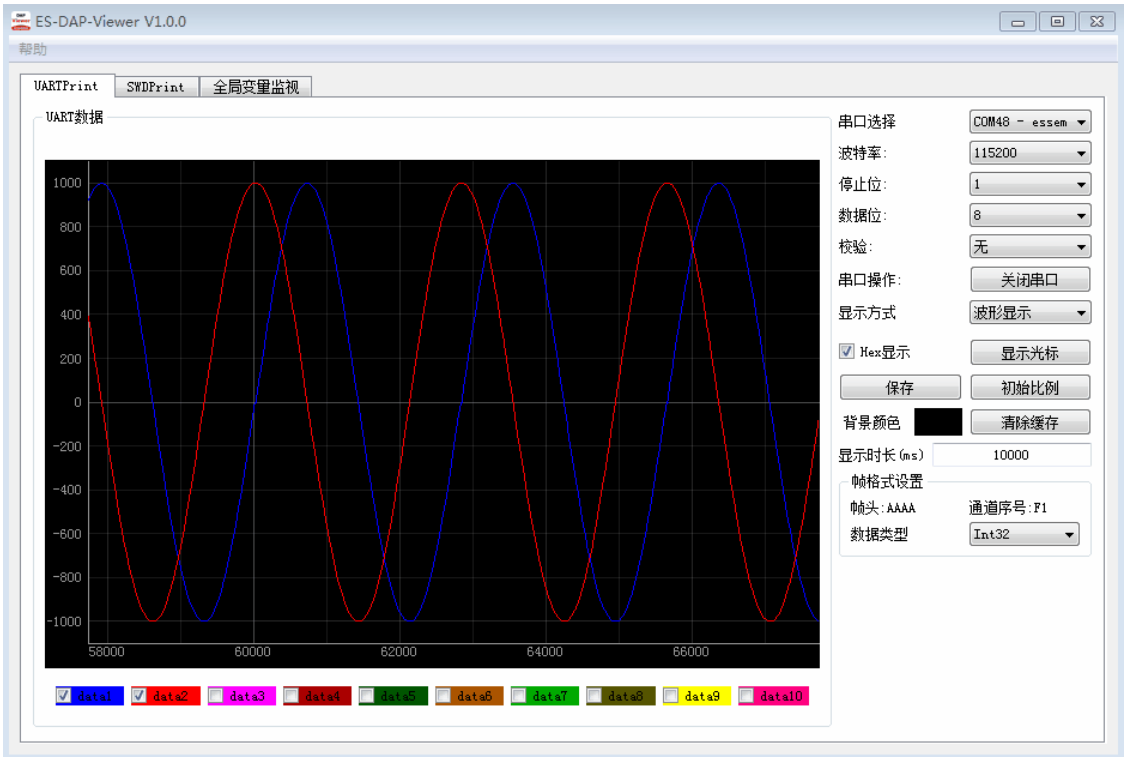

图 3 UART-Print 波形显示界面

左边为绘图窗口；右边为配置窗口，包括串口选择及参数设置窗口，数据处理窗口和帧格式设置窗口；底部为通道选择栏，最多支持 10 个通道。

2.1 PC 界面操作说明

UART-Print 绘图调试的大致流程如下所述：

1. 打开 ES-DAP-Viewer 后选择 UART-Print 选项卡，进入 UART-Print 界面。
2. 在串口选择及参数设置窗口中选择通信串口，并配置通信参数。根据 UART 通信协议，用户需配置波特率，停止位，数据位和奇偶校验位。
3. 在帧格式设置窗口中设置数据类型，可选 Uint8、Int8、Uint16、Int16 以及 Uint32 和 Int32。
4. 点击 打开串口 开始数据传输，点击 开始显示 便可在绘图窗口看到当前调试数据的实时波形。
5. 点击 保存 可保存接收到的数据。点击 清除缓存 可清空当前显示界面内的图形。点击 初始比例 可当前视图回到初始状态时的显示比例。点击 背景颜色 右侧的颜色框可选择当前视图的背景色。

6. 点击 显示方式 下拉菜单，可选择 波形显示 和 数据显示 。需要注意的是，当选择 数据显示 时，在数据显示窗口中显示的是用户程序发送的原始数据（不包含帧头、功能码、数据长度及校验和），每行显示一帧数据。

7. 点击 开始显示，波形或数据开始动态显示。

8. 输入框 显示时长 默认为 50000ms，可根据实际情况，输入合适的值，在按下回车键后将会更新横坐标的显示范围。

2.2 通信协议和程序示例

UART-Print 使用的通信协议如下表所示：

帧头 (2 字节)	功能码 (1 字节)	数据长度 (1 字节)	用户数据 (若干字节)	累加校验和 (1 字节)	备注
0xAAAA	0xF1	data_len	user_data	check_sum	绘图调试功能

表 1 UART-Print 通信协议

关于 ES-DAP-Viewer 的 UART-Print 通信协议，详细说明如下：

1. UART-Print 的数据帧由帧头、功能码、数据长度、用户数据以及累加校验和组成，其中，帧头固定为两字节大小的 0xAAAA，功能码固定为 0xF1。

2. data_len 表示该数据帧内包含的用户数据的字节总长度，不包括帧头、功能码、数据长度和累加校验和。

3. check_sum 表示从该数据帧第一个字节开始，也就是帧头开始，到用户数据最后一个字节的累加校验和，高位舍去，只保留低八位。

4. ES-DAP-Viewer 上位机按照上述的协议解析收到的数据并进行校验以显示可靠的数据，这意味着用户在使用 UART-Print 功能时，必须按照该该协议进行发送数据，否则上位机会因校验不通过而丢弃数据。

5. UART-Print 最多支持 10 个通道，每个通道均支持显示 Uint8、Int8、Uint16、Int16 以及 Uint32 和 Int32 格式的数据。

如下程序可产生上面图片所示的效果。

```
1.
2. int main()
3. {
4.     uint8_t i, j, length;
5.     int32_t data2send[2];
6.     md_uart_init_t g_uart_init;
7.
8.     /* 配置系统时钟 */
```

```
    md_cmu_pll1_config(MD_CMU_PLL1_INPUT_HOSC_3, MD_CMU_PLL1_OUTPUT_72M);
    md_cmu_clock_config(MD_CMU_CLOCK_PLL1, 72000000);
    /* 初始化 SysTick 中断 */
    md_init_1ms_tick();

    /* 使能所有外设时钟 */
    SYSCFG_UNLOCK();
    md_cmu_enable_perh_all();
    SYSCFG_LOCK();

    /* 初始化 UART 引脚 */
    uart_pin_init();

    /* 配置 UART 通信参数 */
    memset(&g_uart_init, 0x0, sizeof(md_uart_init_t));
    g_uart_init.baud        = 115200;
    g_uart_init.word_length = MD_UART_WORD_LENGTH_8B;
    g_uart_init.stop_bits   = MD_UART_STOP_BITS_1;
    g_uart_init.parity      = MD_UART_PARITY_NONE;
    g_uart_init.fctl        = MD_UART_FLOW_CTL_DISABLE;
    md_uart_init(UART0, &g_uart_init);

    i = 0;
    while (1)
    {

        data2send[0] = (int32_t)(1000 * sin((2 * 3.1415926 / 255) * i)); /* 正弦数据 */
        data2send[1] = (int32_t)(1000 * cos((2 * 3.1415926 / 255) * i)); /* 余弦数据 */

        /* 按照 ES-DAP-Viewer 的 UARTPrint 传输协议要求，格式化传输数据，获取待发送数据的总长度
           格式化后的数据暂存在全局变量 g_tx_buf[20] 里，注意数组长度不要越界 */
        length = data_initialize((uint8_t *)data2send, 2 * 4);

        /* 发送格式化后的数据 */
        for (j = 0; j < length; j++)
        {
            md_uart_set_send_data8(UART0, g_tx_buf[j]);
            while (RESET == md_uart_is_active_it_tbc(UART0));
            md_uart_clear_it_tbc(UART0);
```

```
48.         }
49.
50.         i++;
51.         md_delay_1ms(10);
52.     }
53. }
54.
55.
56. int main()
57. {
58.     uint8_t i, j, length;
59.     int32_t data2send[2];
60.     md_uart_init_t g_uart_init;
61.
62.     /* 配置系统时钟 */
63.     md_cmu_pll1_config(MD_CMU_PLL1_INPUT_HOSC_3, MD_CMU_PLL1_OUTPUT_7
    2M);
64.     md_cmu_clock_config(MD_CMU_CLOCK_PLL1, 72000000);
65.     /* 初始化 SysTick 中断 */
66.     md_init_1ms_tick();
67.
68.     /* 使能所有外设时钟 */
69.     SYSCFG_UNLOCK();
70.     md_cmu_enable_perh_all();
71.     SYSCFG_LOCK();
72.
73.     /* 初始化 UART 引脚 */
74.     uart_pin_init();
75.
76.     /* 配置 UART 通信参数 */
77.     memset(&g_uart_init, 0x0, sizeof(md_uart_init_t));
78.     g_uart_init.baud        = 115200;
79.     g_uart_init.word_length = MD_UART_WORD_LENGTH_8B;
80.     g_uart_init.stop_bits   = MD_UART_STOP_BITS_1;
81.     g_uart_init.parity      = MD_UART_PARITY_NONE;
82.     g_uart_init.fctl        = MD_UART_FLOW_CTL_DISABLE;
83.     md_uart_init(UART0, &g_uart_init);
84.
85.     i = 0;
86.     while (1)
87.     {
88.
89.         data2send[0] = (int32_t)(1000 * sin((2 * 3.1415926 / 255) *
    i)); /* 正弦数据 */
```

```
90.         data2send[1] = (int32_t)(1000 * cos((2 * 3.1415926 / 255) *
    i)); /* 余弦数据 */
91.
92.         /* 按照ES-DAP-Viewer 的UARTPrint 传输协议要求，格式化传输数据，获
    取待发送数据的总长度，格式化后的数据暂存在全局变量 g_tx_buf[20] 里，注意数
    组长度不要越界 */
93.         length = data_initialize((uint8_t *)data2send, 2 * 4);
94.
95.         /* 发送格式化后的数据 */
96.         for (j = 0; j < length; j++)
97.         {
98.             md_uart_set_send_data8(UART0, g_tx_buf[j]);
99.             while (RESET == md_uart_is_active_it_tbc(UART0));
100.              md_uart_clear_it_tbc(UART0);
101.          }
102.
103.          i++;
104.          md_delay_1ms(10);
105.      }
106. }
107.
```

第3章 SWD-Print模式

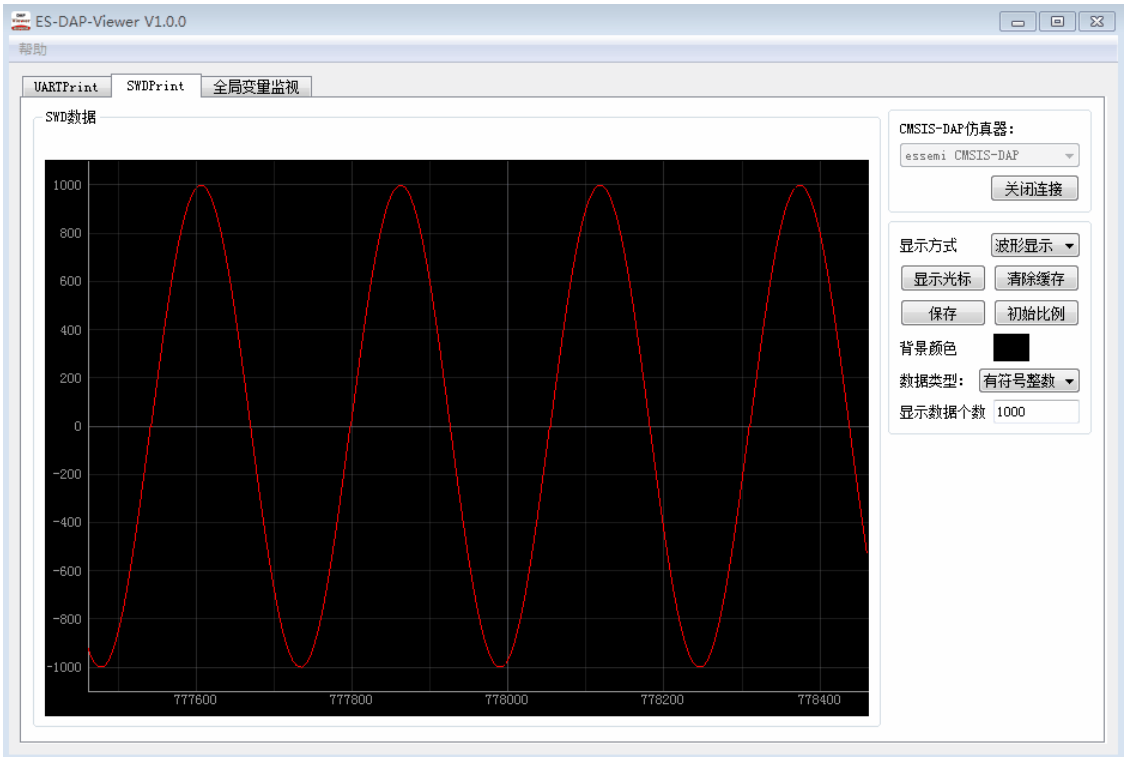

图 4 SWD-Print 波形显示界面

关于 SWD-Print 的界面使用介绍可参考 UART-Print。

3.1 使用说明

用户程序需要使用 essemi_swd_printf 函数将数据写入目标缓冲区，ES-DAP-Viewer 将通过 SWD 调试接口读取数据后显示在 PC 界面上。SWD-Print 功能的实现需要用到以下四个配置文件：

essemi_swd_print.h
essemi_swd_print_conf.h
essemi_swd_print.c
essemi_swd_print_printf.c

用户在工程中成功添加上述四个配置文件后，并在需要用到 SWD-Print 功能的文件中包含如下头文件#include“essemi_swd_print.h”后，即可调用 essemi_swd_printf()函数将需要打印的数据放到 SWD 专属内存区域中，然后上位机到该内存区域取出数据，显示到绘图窗口。

配置文件放在 ES-DAP-Viewer 如下安装目录：...\configure\essemi*swd*print。

需要注意以下几点：

在使用 SWD-Print 功能的函数时，必须先调用函数 int essemi_swd_configupbuffer 以及函数 int essemi_swd_configdownbuffer 来初始化上行（MCU 端到 PC 端）和下行（PC 端到 MCU 端）缓冲区，其中，对于第一个参数 BufferIndex = 0 的时候，SWD 组件已为其配置了缓冲和默认大小，其大小配置是在 essemi_swd_print_conf.h 中通过宏定义进行的 #define

BUFFER_SIZE_UP(1024) 及 #define BUFFER_SIZE_DOWN(16)，因此，在使用缓冲区 0 时，配置比较简单，按照该格式即可：

```
essemi_swd_configupbuffer(0, "SWDUP", NULL, 0, ESSEMI_SWD_MODE_BLOCK_IF_FIFO_FULL);
essemi_swd_configupbuffer(0, "SWDDOWN", NULL, 0, ESSEMI_SWD_MODE_BLOCK_IF_FIFO_FULL);
```

上面两个函数的最后一个参数选择如下宏定义：

```
/* Skip(Default): 如果缓冲区不够存储要发送的数据，将放弃写入缓冲区。*/
#define ESSEMI_SWD_MODE_NO_BLOCK_SKIP   (0)
/* Trim: 如果缓冲区不够存储这些数据，则将无法写入的部分丢弃。*/
#define ESSEMI_SWD_MODE_NO_BLOCK_TRIM   (1)
/* Block: 如果缓冲区满，将阻塞等待有空间可用。*/
#define ESSEMI_SWD_MODE_BLOCK_IF_FIFO_FULL (2)
```

1. SWD-Print 目前仅支持字节、半字、字输出，因此当使用函数 int essemi_swd_printf(unsigned BufferIndex, const char *sFormat, ...) 时，须根据输出的数据位宽使用 “%.2x” 、“%.4x” 或 “%.8x” 来格式化输出。并且，输出的第一个字符必须为空格。如下：

   ```
   essemi_swd_printf(0, " %.8x", data2send[0]); /* 注意：输出的第一个字符必须为空格 */
   ```

2. 在连接 DAP 调试器时，必须确保目标 MCU 程序正确下载并复位，否则将无法连接 DAP 调试器。

3.2 程序示例

如下程序可产生上面图片所示的效果。

```
int main(void)
{
    uint8_t i,j;
    int32_t data2send[2];

    md_cmu_pll1_config(MD_CMU_PLL1_INPUT_HOSC_3, MD_CMU_PLL1_OUTPUT_96M);
    md_cmu_clock_config(MD_CMU_CLOCK_PLL1, 96000000);

    /* Enable ALL peripheral */
    SYSCFG_UNLOCK();
    md_cmu_enable_perh_all();
    SYSCFG_LOCK();

    /* 初始化缓冲区 */
```

```
    essemi_swd_configupbuffer(0, "SWDUP", NULL, 0, ESSEMI_SWD_MODE_BLOCK_IF_FIFO_FULL);
    essemi_swd_configdownbuffer(0, "SWDDOWN", NULL, 0, ESSEMI_SWD_MODE_BLOCK_IF_FIFO_FULL);

    i = 0;
    while (1)
    {
        data2send[0] = (int32_t)(1000 * sin((2 * 3.1415926 / 255) * i));
        essemi_swd_printf(0, " %.8x", data2send[0]); /* 向SWD端口缓冲区输出数据，注意输出的第一个字符必须为空格，并根据数据位宽将数据格式输出 */
        i++;
    }
}

```

第4章　全局变量监视模式

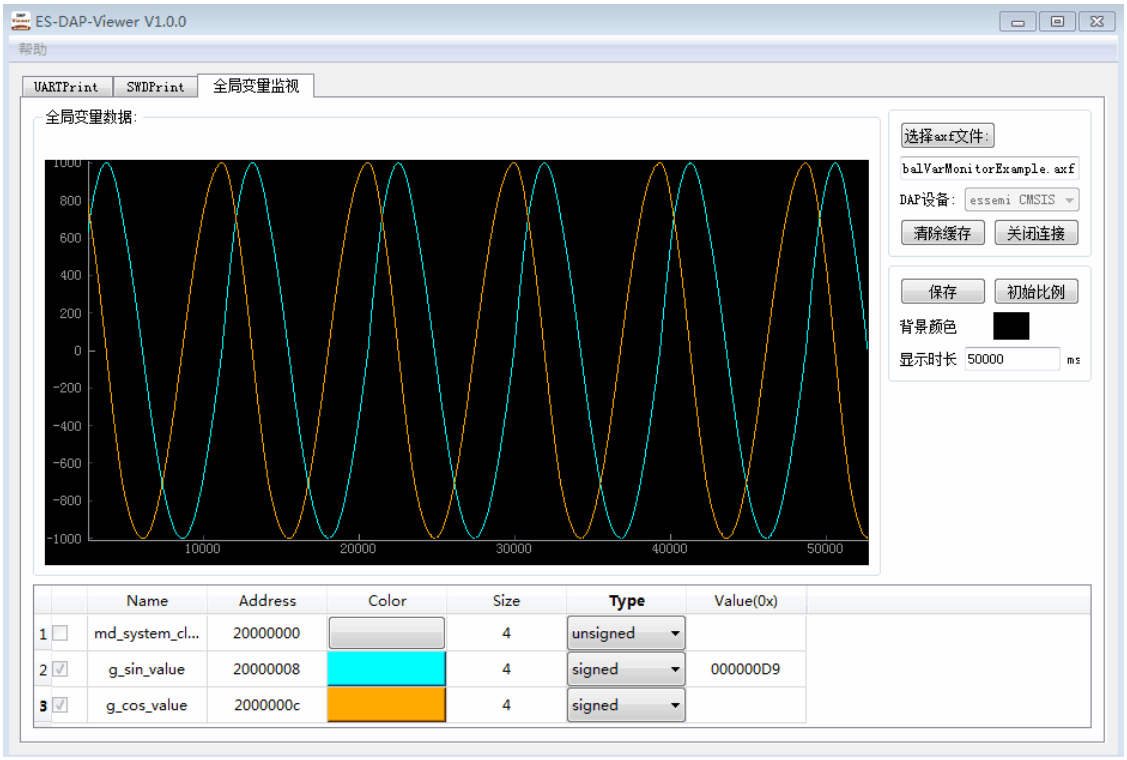

图 5　全局变量监视调试界面

4.1　使用说明

用户首先需点击 选择 axf 文件 按钮，载入由编译器生成的 axf 文件，然后在页面下方选择需要监视的全局变量。

通过点击下拉框 DAP 设备 选择上位机已扫描到的 DAP 设备，当上位机扫描到目标 CMSIS-DAP 设备时，点击 打开连接 即可与该设备建立通信。

左边为绘图窗口；右边为配置窗口，包括 axf 文件和 DAP 设备选择窗口，数据显示配置窗口；下方为解析到的全局变量列表。

全局变量监视绘图调试的流程如下所述：

1. 打开 ES-DAP-Viewer 后选择全局变量监视选项卡，进入全局变量监视界面。

2. 点击 选择 axf 文件 按钮，载入由编译器生成的 axf 文件。

3. 点击下拉框 DAP 设备 选择上位机已扫描到的 DAP 设备，当上位机扫描到目标 DAP 设备时，点击 打开连接 与该设备建立连接。

4. 在下方全局变量列表中，可查看解析到的全局变量的名称，地址以及数据类型大小，通过点击全局变量所在行中，对应的 Color 单元，可选择该全局变量在波形显示窗口中的颜色。此外，尽管目前可以解析到结构体变量，但目前不支持对结构体成员和数组成员进行解析，因此在全局变量列表中勾选结构体类型的全局变量是无效的，勾选数组成员会因数组类型的不同而产生不同的效果。

5. 点击 开始显示 按钮便可在绘图窗口看到当前调试数据的实时变化波形，在下方全局变量列表的 Value(0x) 列中可查看全局变量的当前值。

6. 此外，用户还可在数据处理窗口中选择保存数据，显示光标，清除缓存，以及更改背景颜色。

7. 其余操作可参考 UART-Print 界面操作说明。

4.2 程序示例

如下程序可产生上面图片所示的效果。

```
1.
2.  int main(void)
3.  {
4.      uint16_t i;
5.      md_cmu_pll1_config(MD_CMU_PLL1_INPUT_HOSC_3,
    MD_CMU_PLL1_OUTPUT_96M);
6.      md_cmu_clock_config(MD_CMU_CLOCK_PLL1, 96000000);
7.
8.      /* Enable ALL peripheral */
9.      SYSCFG_UNLOCK();
10.     md_cmu_enable_perh_all();
11.     SYSCFG_LOCK();
12.
13.     i = 0;
14.     while (1)
15.     {
16.         g_sin_value = (int32_t)(1000 * sin((2 * 3.1415926 / 65535) * i));
    /* 产生正弦数据 */
17.         g_cos_value = (int32_t)(1000 * cos((2 * 3.1415926 / 65535) * i));
    /* 产生余弦数据 */
18.
19.         i++;
20.     }
21. }
22.
23.
```