CONSTRUCTION AND TESTING FOR HEADER RULES COMPLIANCE (Kieras, 2012)

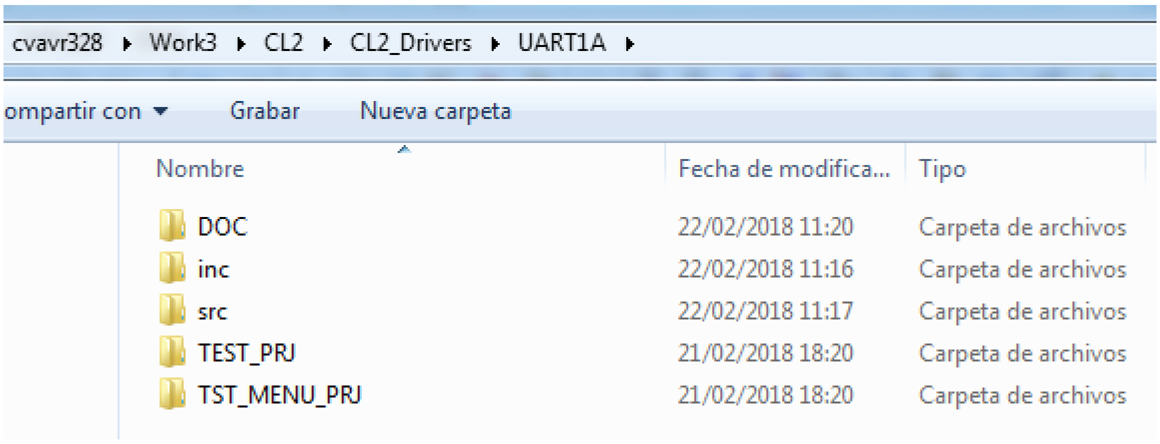
UART1A/UART1_DR1 MODULE on CVAVR 3 / CL2 – Rev. 21-02-2018 / R.Oliva

(UART1A / Variable Init Baud Rate, to use 4DSys Displays) Updated with test results 22-02-2018

Reformatted – printout in PDF form

CREATION 21-02-2018 – UART1_DR1 - (Files UART1_DR1.c / .h, formerly within mainfile)
This document in: "C:\cvavr328\Work3\CL2\CL2_Drivers\UART1A\DOC\UART1A_DR1_CONSTR+TESTING FOR HEADER RULES COMPLIANCE_v21-02-2018.docx"

A) Directory structure proposed:



Nombre	Fecha de modifica...	Tipo
DOC	22/02/2018 11:20	Carpeta de archivos
inc	22/02/2018 11:16	Carpeta de archivos
src	22/02/2018 11:17	Carpeta de archivos
TEST_PRJ	21/02/2018 18:20	Carpeta de archivos
TST_MENU_PRJ	21/02/2018 18:20	Carpeta de archivos

B) Version Notes: 22.02.2018 This UART1A/UART1_DR1 version adds the possibility of selecting a parameter for different baud-rates within the UART1_Init(parm) function. The possible baud rates at this version are 9600 (for 4D sytems), 19200 and 38400 (used in METEO sytems). Table 1 is taken from the ATMega1284P manual, and shows selection of UBRR1 register settings for these options, which are set in the new function call:

```
void USART1_Init(uint8_t pbaud);
```

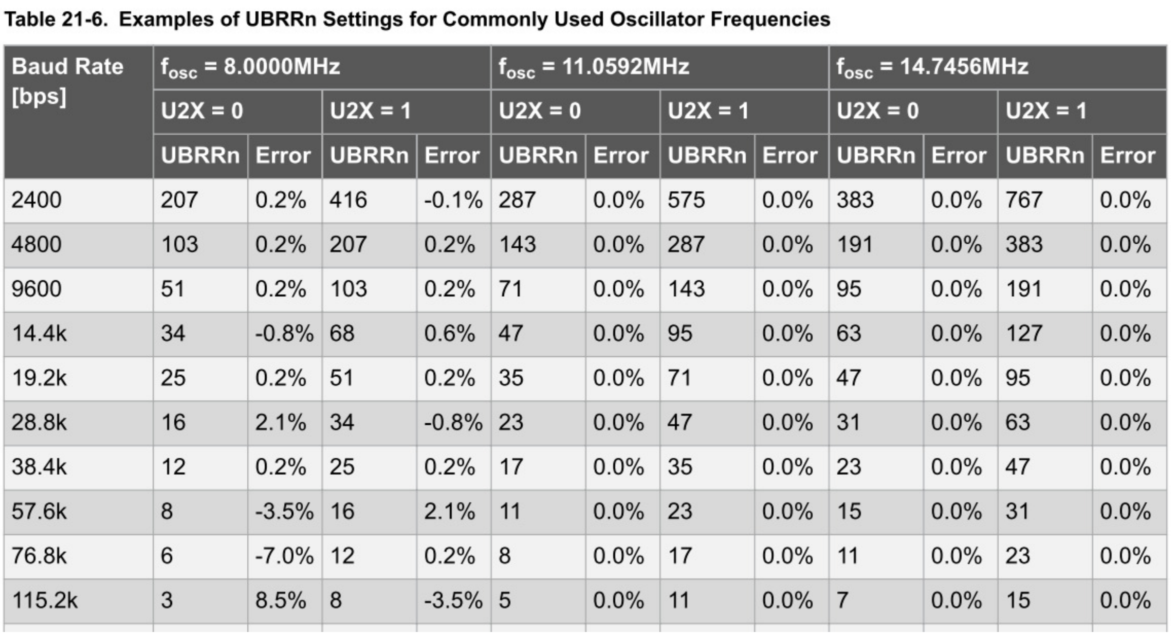


Table 21-6. Examples of UBRRn Settings for Commonly Used Oscillator Frequencies

Baud Rate [bps]	f_{osc} = 8.0000MHz				f_{osc} = 11.0592MHz				f_{osc} = 14.7456MHz			
	U2X = 0		U2X = 1		U2X = 0		U2X = 1		U2X = 0		U2X = 1	
	UBRRn	Error	UBRRn	Error	UBRRn	Error	UBRRn	Error	UBRRn	Error	UBRRn	Error
2400	207	0.2%	416	-0.1%	287	0.0%	575	0.0%	383	0.0%	767	0.0%
4800	103	0.2%	207	0.2%	143	0.0%	287	0.0%	191	0.0%	383	0.0%
9600	51	0.2%	103	0.2%	71	0.0%	143	0.0%	95	0.0%	191	0.0%
14.4k	34	-0.8%	68	0.6%	47	0.0%	95	0.0%	63	0.0%	127	0.0%
19.2k	25	0.2%	51	0.2%	35	0.0%	71	0.0%	47	0.0%	95	0.0%
28.8k	16	2.1%	34	-0.8%	23	0.0%	47	0.0%	31	0.0%	63	0.0%
38.4k	12	0.2%	25	0.2%	17	0.0%	35	0.0%	23	0.0%	47	0.0%
57.6k	8	-3.5%	16	2.1%	11	0.0%	23	0.0%	15	0.0%	31	0.0%
76.8k	6	-7.0%	12	0.2%	8	0.0%	17	0.0%	11	0.0%	23	0.0%
115.2k	3	8.5%	8	-3.5%	5	0.0%	11	0.0%	7	0.0%	15	0.0%

Table 1 - Selection of possible values from ATMega1284P manual.

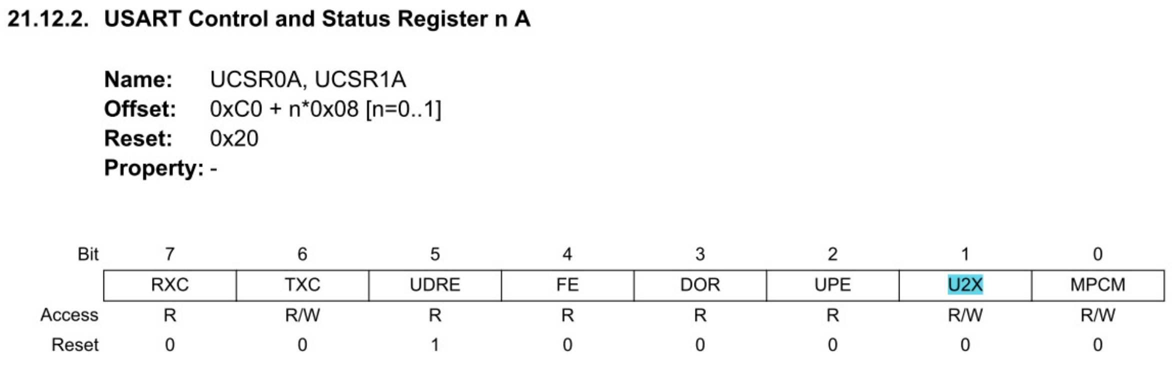
In our case, Baud Rates will be selectable:

38400 -> u2x=0 ->UBRR=23dec = 0x17

19200 -> u2x=0 ->UBRR=47dec = 0x2F

9600 -> u2x=0 ->UBRR=95dec = 0x5F

The duplicating rate U2X bit in UCSR1A, will be set to 0 in this version. From the manual, this bit is:



21.12.2. USART Control and Status Register n A

Name: UCSR0A, UCSR1A
Offset: 0xC0 + n*0x08 [n=0..1]
Reset: 0x20
Property: -

Bit	7	6	5	4	3	2	1	0
	RXC	TXC	UDRE	FE	DOR	UPE	U2X	MPCM
Access	R	R/W	R	R	R	R	R/W	R/W
Reset	0	0	1	0	0	0	0	0

C) Kieras Rule checking.

Rule #1 – OK Groups functional operations with UART1

Rule #2 OK – “Include guards”, used here:

```
#ifndef UART1_INCLUDED
#define UART1_INCLUDED
…
#endif
```

Rule #3 All required declarations to use the module appear in the UART1_DR1.h file

Rule #4 OK → .h file only contains declarations, and is included by the .c file

```
.C FILE:
// ***********************************************************
// local functions
// ***********************************************************

NO LOCAL FUNCTIONS IN .C FILE
```

At beginning includes its header: #include "../inc/UART1_DR1.h" (OK)

```
.H FILE:
```

At start of .h file, these are published functions for all-prgrm access:

```
/************************************************************************
**
**    Functions public to rest of program
**
****************************************************************************/

// Interrupt routines
interrupt [USART1_RXC] void usart1_rx_isr(void);
interrupt [USART1_TXC] void usart1_tx_isr(void);

// Alternate getchar1() defined with ISR Rx support
char getchar1(void);

// Alternate putchar1() defined with ISR Tx support
void putchar1(char c);

// USART1_Init modified (UART1A) for pbaud parameter 21.2.18
void USART1_Init(uint8_t pbaud);
```

```
// New Functions defined in .C file:
/*****************************************************************************
**
** Initializes the UART1 - version UART1A with parameter 21.2.18
**
** Parameters: uint8_t pbaud, can take values:
** #define PBAUD_9600  0
** #define PBAUD_19200 1
** #define PBAUD_38400 2
** Then for each option, considering CLK=14.7456E06 Hz
**   38400 -> u2x=0 ->UBRR=23dec = 0x17
**  19200 -> u2x=0 ->UBRR=47dec = 0x2F
**  9600  -> u2x=0 ->UBRR=95dec = 0x5F
**
** Returns: NONE
**
*****************************************************************************/
// USART1_Init modified for pbaud parameter
void USART1_Init(uint8_t pbaud)
{
  switch(pbaud){
        case PBAUD_9600:
                // USART1 initialization 9600 baud
                // Communication Parameters: 8 Data, 1 Stop, No Parity
                // USART1 Receiver: On
                // USART1 Transmitter: On
                // USART1 Mode: Asynchronous
                // USART1 Baud Rate: 9600
                UCSR1A=0x00;
                UCSR1B=0xD8;
                UCSR1C=0x06;
                UBRR1H=0x00;
                UBRR1L=0x5F; // ==95 dec for 9600, U2X=0
                break;
        case PBAUD_19200:
                // USART1 initialization 19200
                // Communication Parameters: 8 Data, 1 Stop, No Parity
                // USART1 Receiver: On
                // USART1 Transmitter: On
                // USART1 Mode: Asynchronous
                // USART1 Baud Rate: 19200
                UCSR1A=0x00;
                UCSR1B=0xD8;
                UCSR1C=0x06;
                UBRR1H=0x00;
                UBRR1L=0x2F; // ==47 dec for 19200, U2X=0
                break;
        case PBAUD_38400:
                // USART1 initialization 38400 baud (PWRC2 - V22.3.2012)
                // Communication Parameters: 8 Data, 1 Stop, No Parity
                // USART1 Receiver: On
                // USART1 Transmitter: On
                // USART1 Mode: Asynchronous
                // USART1 Baud Rate: 38400
                UCSR1A=0x00;
                UCSR1B=0xD8;
                UCSR1C=0x06;
                UBRR1H=0x00;
                UBRR1L=0x17; // ==23 dec for 38400, U2X=0
                break;
        default:
        printf("\n\r Parametro COM1 Incorrecto! (9600, 19200 o 38400)");
        break;
        }
}


// Slightly modified RX_ISR:
/******************************************************************************
**
** USART1 Receiver interrupt service routine
** Buffer Size 256 not considered..30.1.18
******************************************************************************/

interrupt [USART1_RXC] void usart1_rx_isr(void)
{
char status,data;
status=UCSR1A;
data=UDR1;
```

```
if ((status & (FRAMING_ERROR | PARITY_ERROR | DATA_OVERRUN))==0)
   {
   rx_buffer1[rx_wr_index1++]=data;
   // #if RX_BUFFER_SIZE1 == 256
   // special case for receiver buffer size=256
   // if (++rx_counter1 == 0)
   //    {
   // #else
   if (rx_wr_index1 == RX_BUFFER_SIZE1) rx_wr_index1=0;
   if (++rx_counter1 == RX_BUFFER_SIZE1)
      {
      rx_counter1=0;
   //#endif
      rx_buffer_overflow1=1;
      }
   }
}
```

Rule #5 OK → Globally used variables are declared as extern in .h file, and defined in .c file:

```
/ /****************************************************************************
**
**    EXPORTED VARIABLES
**    declared here, but defined in .c file for global access.. 30.01.2018
**
*****************************************************************************/

extern char rx_buffer1[RX_BUFFER_SIZE1];


#if RX_BUFFER_SIZE1 <= 256
extern unsigned char rx_wr_index1,rx_rd_index1,rx_counter1;
#else
extern unsigned int rx_wr_index1,rx_rd_index1,rx_counter1;
#endif

// This flag is set on USART1 Receiver buffer overflow
extern bit rx_buffer_overflow1;

extern char tx_buffer1[TX_BUFFER_SIZE1];

#if TX_BUFFER_SIZE1 <= 256
extern unsigned char tx_wr_index1,tx_rd_index1,tx_counter1;
#else
extern unsigned int tx_wr_index1,tx_rd_index1,tx_counter1;
#endif
```

These VARIABLES are memory assigned in the .c file as follows:

```
/*****************************************************************************
**
** UART1 Global Variables declared in uart1_dr1.h
** MEMORY IS ASSIGNED
** HERE FOLLOWING RULE #5
**
*****************************************************************************/

char rx_buffer1[RX_BUFFER_SIZE1];

#if RX_BUFFER_SIZE1 <= 256
unsigned char rx_wr_index1,rx_rd_index1,rx_counter1;
#else
unsigned int rx_wr_index1,rx_rd_index1,rx_counter1;
#endif

// This flag is set on USART1 Receiver buffer overflow
bit rx_buffer_overflow1;

char tx_buffer1[TX_BUFFER_SIZE1];

#if TX_BUFFER_SIZE1 <= 256
unsigned char tx_wr_index1,tx_rd_index1,tx_counter1;
#else
unsigned int tx_wr_index1,tx_rd_index1,tx_counter1;
#endif
```

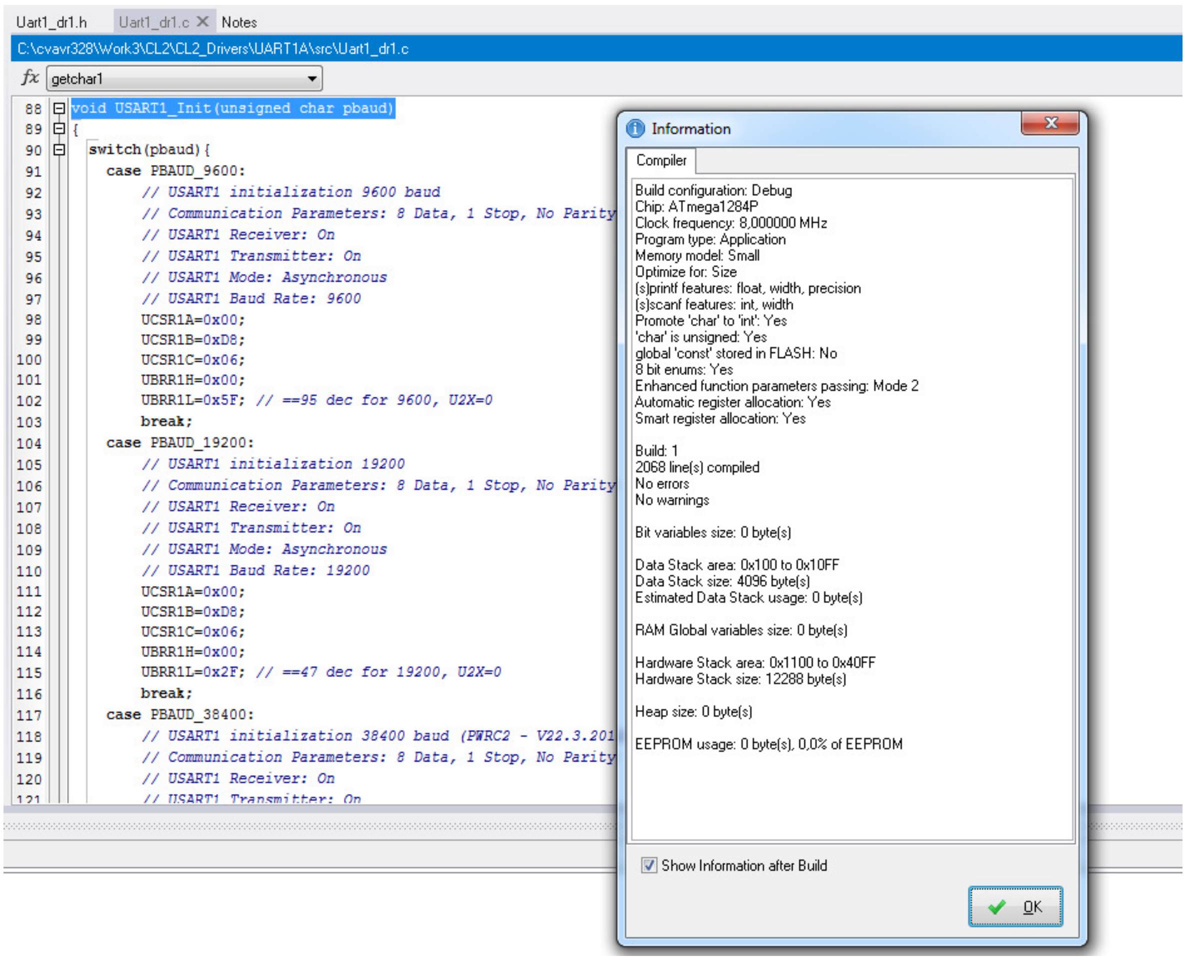
Rule #6 Internal declarations kept out of .h module→ Ok

Rule #7,8 Not Applicable, since no external .h functions are required for this module.

Rule #9 Self compilation :

In CVAVR 3 we need to make a Test_Uart1Dr.prj, including only the file uart1_dr1.c, which at the start executes: #include "../inc/ uart1_dr1.h" – see if it compiles correctly by itself.

> PRJ file should be confined it to:
> C:\cvavr328\Work3\CL2\CL2_Drivers\. It creates a test_dr.c empty file, which is not used at this stage..


Uart1_dr1.h
Uart1_dr1.c
Notes
C:\cvavr328\Work3\CL2\CL2_Drivers\UART1A\src\Uart1_dr1.c
getchar1
Information
Compiler
Show Information after Build
OK

```
void USART1_Init(unsigned char pbaud)
{
  switch(pbaud) {
    case PBAUD_9600:
        // USART1 initialization 9600 baud
        // Communication Parameters: 8 Data, 1 Stop, No Parity
        // USART1 Receiver: On
        // USART1 Transmitter: On
        // USART1 Mode: Asynchronous
        // USART1 Baud Rate: 9600
        UCSR1A=0x00;
        UCSR1B=0xD8;
        UCSR1C=0x06;
        UBRR1H=0x00;
        UBRR1L=0x5F; // ==95 dec for 9600, U2X=0
        break;
    case PBAUD_19200:
        // USART1 initialization 19200
        // Communication Parameters: 8 Data, 1 Stop, No Parity
        // USART1 Receiver: On
        // USART1 Transmitter: On
        // USART1 Mode: Asynchronous
        // USART1 Baud Rate: 19200
        UCSR1A=0x00;
        UCSR1B=0xD8;
        UCSR1C=0x06;
        UBRR1H=0x00;
        UBRR1L=0x2F; // ==47 dec for 19200, U2X=0
        break;
    case PBAUD_38400:
        // USART1 initialization 38400 baud (PWRC2 - V22.3.201
        // Communication Parameters: 8 Data, 1 Stop, No Parity
        // USART1 Receiver: On
        // USART1 Transmitter: On
```

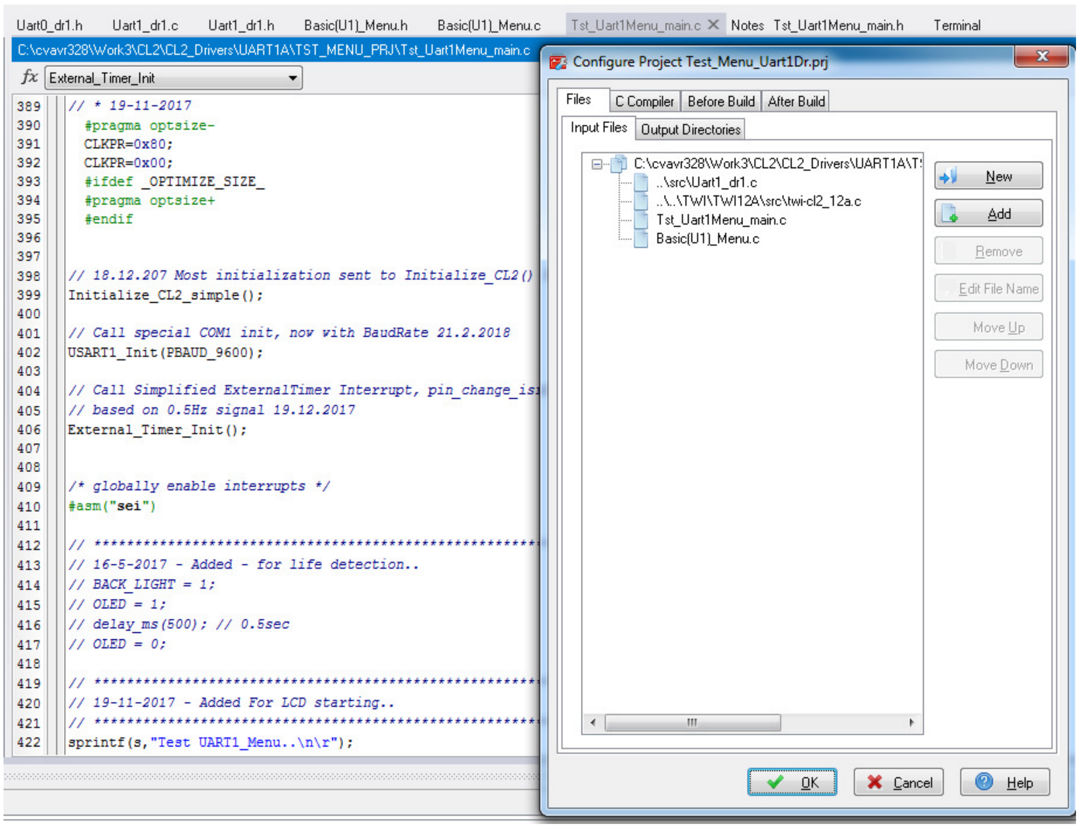
```
Build configuration: Debug
Chip: ATmega1284P
Clock frequency: 8,000000 MHz
Program type: Application
Memory model: Small
Optimize for: Size
(s)printf features: float, width, precision
(s)scanf features: int, width
Promote 'char' to 'int': Yes
'char' is unsigned: Yes
global 'const' stored in FLASH: No
8 bit enums: Yes
Enhanced function parameters passing: Mode 2
Automatic register allocation: Yes
Smart register allocation: Yes

Build: 1
2068 line(s) compiled
No errors
No warnings

Bit variables size: 0 byte(s)

Data Stack area: 0x100 to 0x10FF
Data Stack size: 4096 byte(s)
Estimated Data Stack usage: 0 byte(s)

RAM Global variables size: 0 byte(s)

Hardware Stack area: 0x1100 to 0x40FF
Hardware Stack size: 12288 byte(s)

Heap size: 0 byte(s)

EEPROM usage: 0 byte(s), 0,0% of EEPROM
```

Rule #10 OK: UART1_DR1.c includes at beginning the file UART1_DR1.h, other files not applicable, since no external .h functions are required for this module.

Rule #11 OK no .c files #included.

TESTING:

A) 22.02.2018 NOW TEST NEW UART1A/UART1_DR1 WITH TST_MENU_UART1.PRJ,
C:\cvavr328\Work3\CL2\CL2_Drivers\UART1A\TST_FUNC_PRJ\Test_Menu_Uart1Dr.prj
This Project copies (A) and includes a simple Menu System borrowed from PWRC2, to test the reading & writing via UART1 drivers, specially at 9600 baud (required by 4DSystems Displays).

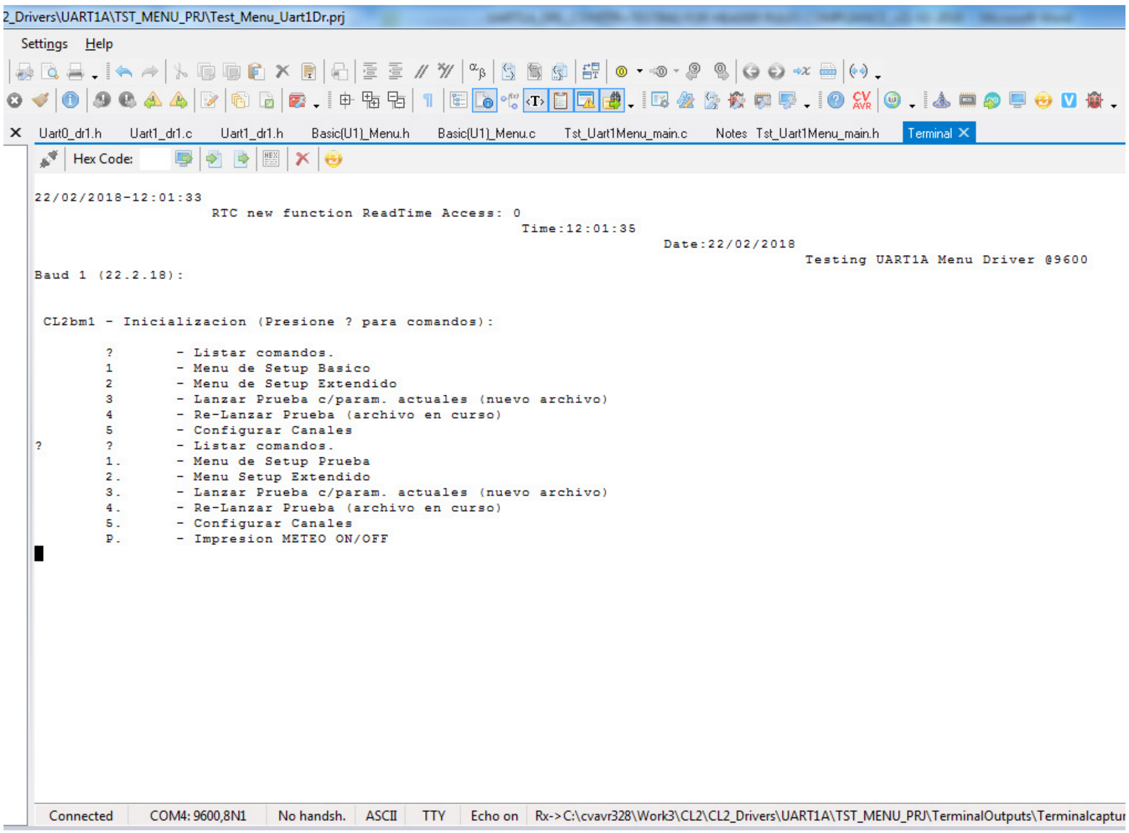


```
// * 19-11-2017
  #pragma optsize-
  CLKPR=0x80;
  CLKPR=0x00;
  #ifdef _OPTIMIZE_SIZE_
  #pragma optsize+
  #endif


// 18.12.207 Most initialization sent to Initialize_CL2()
Initialize_CL2_simple();

// Call special COM1 init, now with BaudRate 21.2.2018
USART1_Init(PBAUD_9600);

// Call Simplified ExternalTimer Interrupt, pin_change_is
// based on 0.5Hz signal 19.12.2017
External_Timer_Init();


/* globally enable interrupts */
#asm("sei")

// ******************************************************
// 16-5-2017 - Added - for life detection..
// BACK_LIGHT = 1;
// OLED = 1;
// delay_ms(500); // 0.5sec
// OLED = 0;

// ******************************************************
// 19-11-2017 - Added For LCD starting..
// ******************************************************
sprintf(s,"Test UART1_Menu..\n\r");
```

```
C:\cvavr328\Work3\CL2\CL2_Drivers\UART1A\T
    ..\src\Uart1_dr1.c
    ..\..\TWI\TWI12A\src\twi-cl2_12a.c
    Tst_Uart1Menu_main.c
    Basic(U1)_Menu.c
```

SOURCE CODE For Menu on UART1 / UART1A (see project)

We set up terminal for testing, 9600,n1,8,1 – works ok in transmission & reception..:

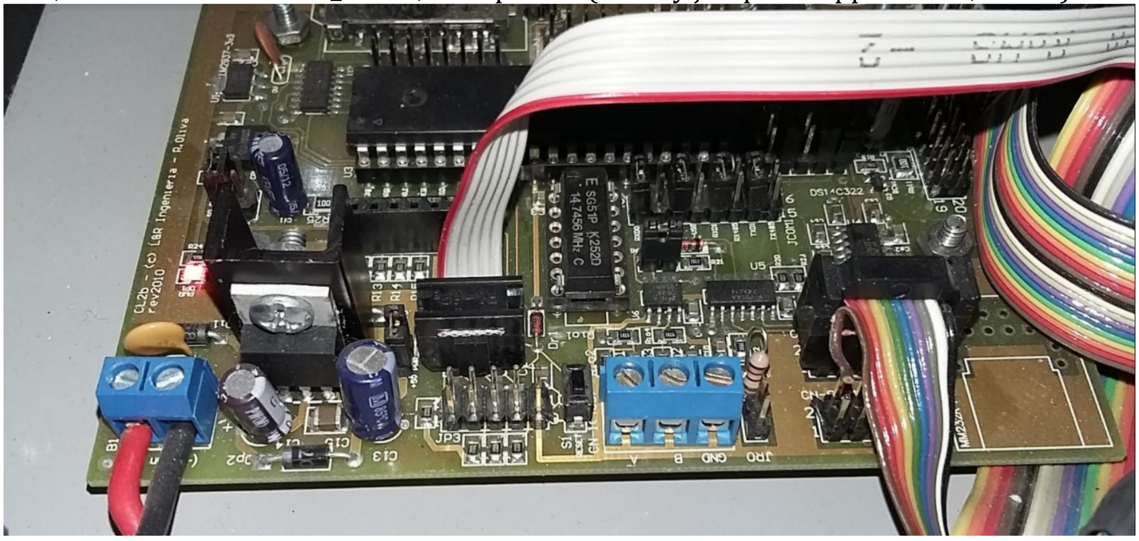


```
22/02/2018-12:01:33
                         RTC new function ReadTime Access: 0
                                                            Time:12:01:35
                                                                              Date:22/02/2018
                                                                                              Testing UART1A Menu Driver @9600
Baud 1 (22.2.18):


 CL2bm1 - Inicializacion (Presione ? para comandos):

        ?       - Listar comandos.
        1       - Menu de Setup Basico
        2       - Menu de Setup Extendido
        3       - Lanzar Prueba c/param. actuales (nuevo archivo)
        4       - Re-Lanzar Prueba (archivo en curso)
        5       - Configurar Canales
?       ?       - Listar comandos.
        1.      - Menu de Setup Prueba
        2.      - Menu Setup Extendido
        3.      - Lanzar Prueba c/param. actuales (nuevo archivo)
        4.      - Re-Lanzar Prueba (archivo en curso)
        5.      - Configurar Canales
        P.      - Impresion METEO ON/OFF
█
```

Connected | COM4: 9600,8N1 | No handsh. | ASCII | TTY | Echo on | Rx->C:\cvavr328\Work3\CL2\CL2_Drivers\UART1A\TST_MENU_PRJ\TerminalOutputs\Terminalcaptur

Seems to work alright!

RX1, TX1 are routed to CN_COM2, as in photo.. (usually jumper-mapped to RX,TX485)

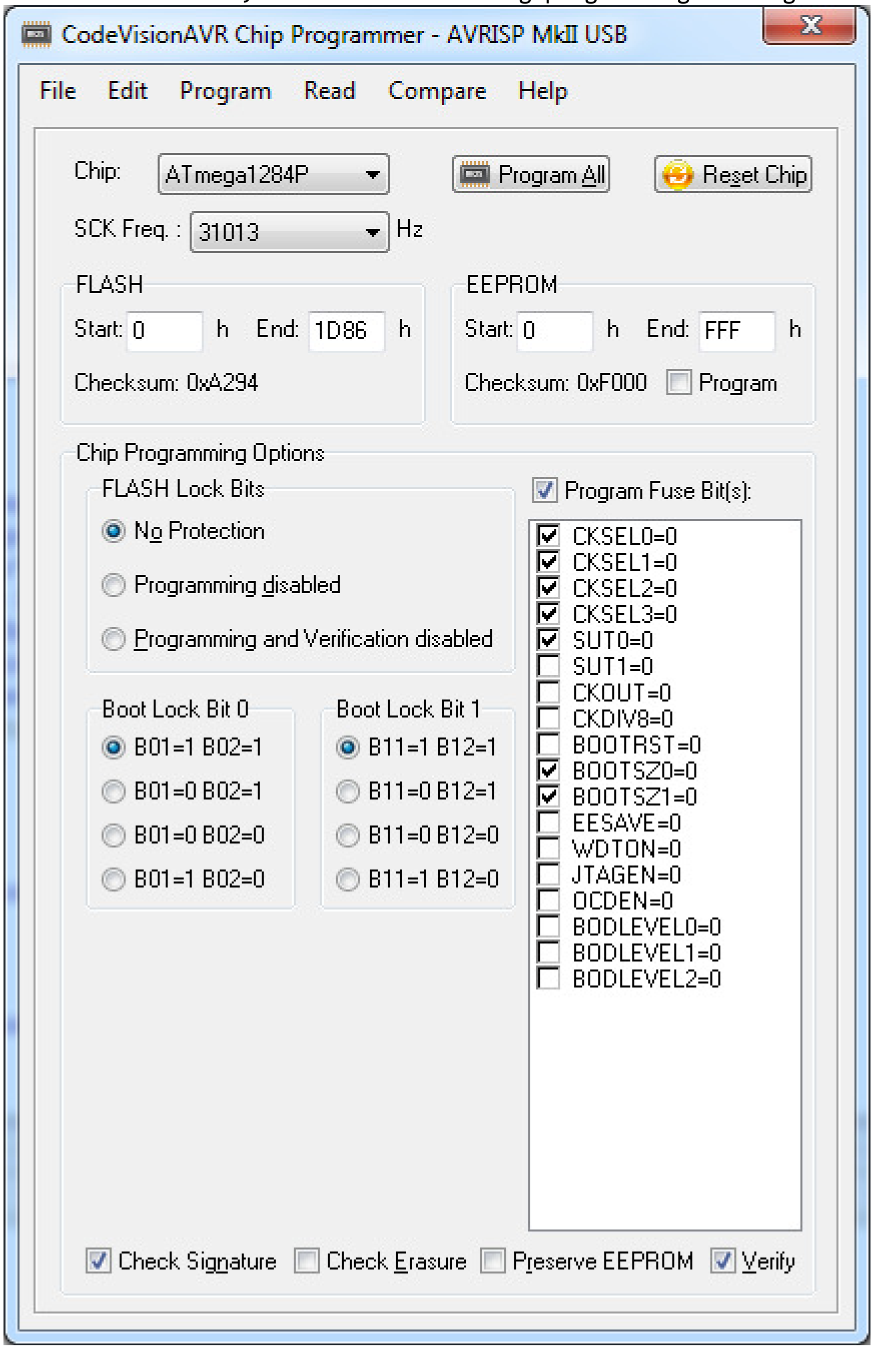


New .c/.h test files are:
`Basic(U1)_Menu.c / .h`

```
TEST_MENU_Uart1Dr.PRJ PROJECT LOOKS LIKE THIS: (like a visual MAKE) on CVAVR3
  As seen: We have a main Tst_UartMenu1_main.c file, which contains main(). An annex file called
    Basic(U1)_Menu.c contains the UART1_Menu() routine (from PWRC2, simplified) and auxiliary
   functions. Also, the two driver functions (UART1A, and TWI-CL2A ) are called within the PRJ
```

Project of new UART1_dr1.c/.h Menu testing..

APPENDIX: As usual, we use the following programming settings:



CodeVisionAVR Chip Programmer - AVRISP MkII USB

File Edit Program Read Compare Help

Chip: ATmega1284P   Program All   Reset Chip

SCK Freq. : 31013 Hz

FLASH
Start: 0 h End: 1D86 h
Checksum: 0xA294

EEPROM
Start: 0 h End: FFF h
Checksum: 0xF000 ☐ Program

Chip Programming Options

FLASH Lock Bits
● No Protection
○ Programming disabled
○ Programming and Verification disabled

Boot Lock Bit 0
● B01=1 B02=1
○ B01=0 B02=1
○ B01=0 B02=0
○ B01=1 B02=0

Boot Lock Bit 1
● B11=1 B12=1
○ B11=0 B12=1
○ B11=0 B12=0
○ B11=1 B12=0

☑ Program Fuse Bit(s):
☑ CKSEL0=0
☑ CKSEL1=0
☑ CKSEL2=0
☑ CKSEL3=0
☑ SUT0=0
☐ SUT1=0
☐ CKOUT=0
☐ CKDIV8=0
☐ BOOTRST=0
☑ BOOTSZ0=0
☑ BOOTSZ1=0
☐ EESAVE=0
☐ WDTON=0
☐ JTAGEN=0
☐ OCDEN=0
☐ BODLEVEL0=0
☐ BODLEVEL1=0
☐ BODLEVEL2=0

☑ Check Signature ☐ Check Erasure ☐ Preserve EEPROM ☑ Verify