King Mongkut's University of Technology Thonburi

Department of Electronics and Telecommunication Engineering Faculty of Engineering

EIE/ENE 335 Digital Circuit and Microprocessor Lab for the 3rd year student

Experiment: Universal Asynchronous Receiver/Transmitters (UART)

Objectives

- How to use
  - the NuMicro™ NUC100 series driver to do the fast application software development
  - UART

Background Theory

UART

NuMicro™ NUC130/NUC140 provides up to three channels of Universal Asynchronous Receiver/Transmitters (UART). UART0 supports High Speed UART and UART1~2 perform Normal Speed UART, besides, only UART0 and UART1 support flow control function.

The Universal Asynchronous Receiver/Transmitter (UART) performs a serial-to-parallel conversion on data received from the peripheral, and a parallel-to-serial conversion on data transmitted from the CPU. The UART controller also supports IrDA SIR Function, LIN master/slave mode function and RS-485 mode functions. Each UART channel supports seven types of interrupts.

The UART0 is built-in with a 64-byte transmitter FIFO (TX_FIFO) and a 64-byte receiver FIFO (RX_FIFO) that reduces the number of interrupts presented to the CPU and the UART1~2 are equipped 16-byte transmitter FIFO (TX_FIFO) and 16-byte receiver FIFO (RX_FIFO). The CPU can read the status of the UART at any time during the operation. The reported status information includes the type and condition of the transfer operations being performed by the UART, as well as 4 error conditions. The UART includes a programmable baud rate generator that is capable of dividing clock input by divisors to produce the serial clock that transmitter and receiver need. The baud rate equation is Baud Rate = UART_CLK / M * [BRD + 2], where M and BRD are defined in Baud Rate Divider Register (UA_BAUD).

Mode	DIV_X_EN	DIV_X_ONE	Divider X	BRD	Baud rate equation
0	0	0	B	A	UART_CLK / [16 * (A+2)]
1	1	0	B	A	UART_CLK / [(B+1) * (A+2)] , B must >= 8
2	1	1	Don't care	A	UART_CLK / (A+2), A must >=3

Figure 1 UART Baud Rate Equation

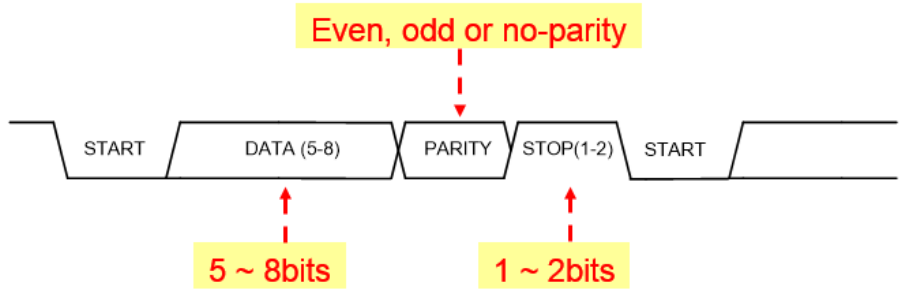


Figure 2 UART Timing

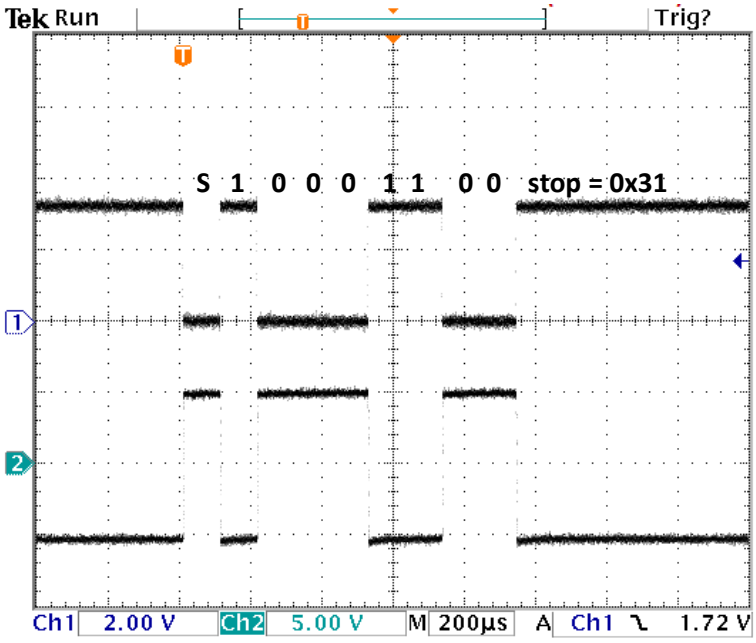


Figure 3 UART: databit-8, stop-1, parity-none

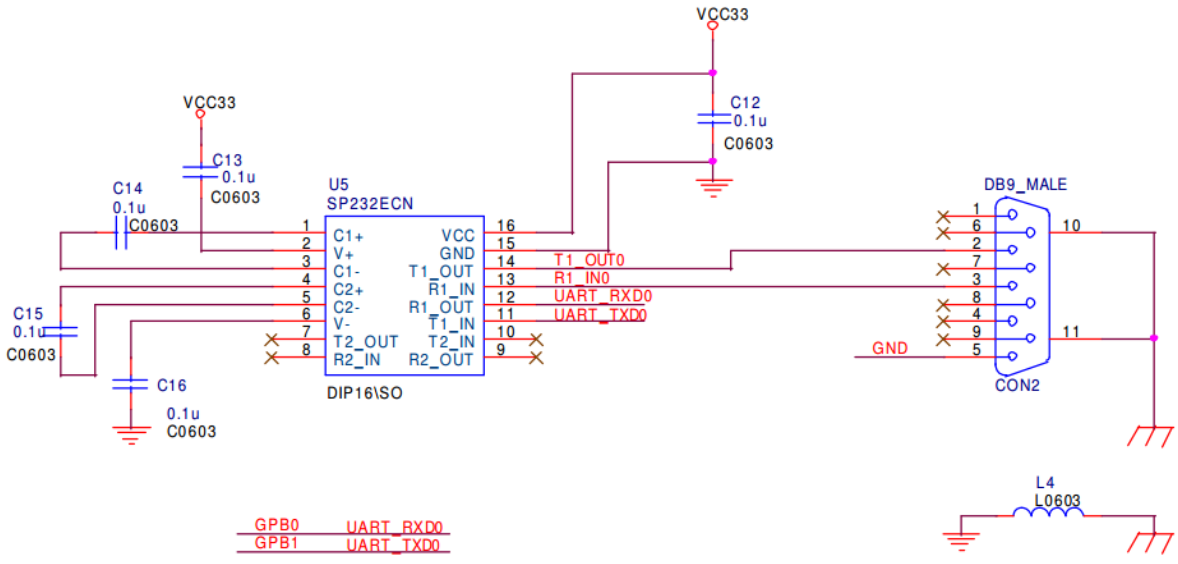


Figure 4 UART0 Schematic

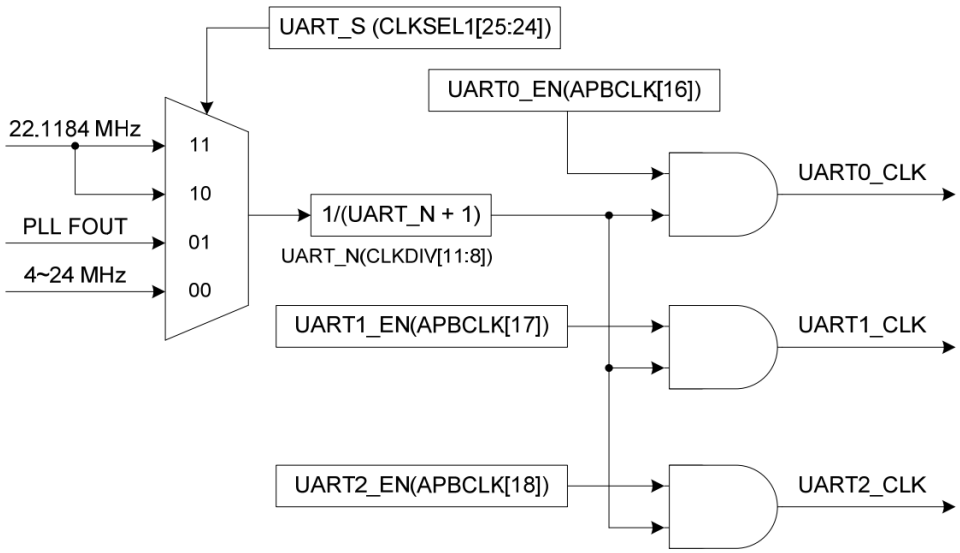
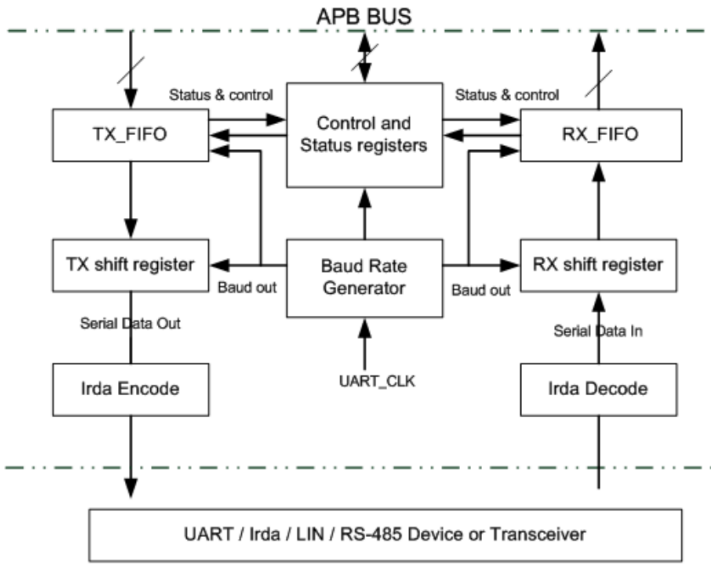


Figure 5 UART Clock Control Diagram



Note : UART0 equips 64 bytes FIFO, UART1/UART2 equip 16 bytes FIFO

Figure 6 UART Clock Control Diagram

	RS232	RS423	RS422	RS485
Differential	no	no	yes	yes
Max number of drivers Max number of receivers	1 1	1 10	1 10	32 32
Modes of operation	half duplex full duplex	half duplex	half duplex	half duplex
Network topology	point-to-point	multidrop	multidrop	multipoint
Max distance (acc. standard)	15 m	1200 m	1200 m	1200 m
Max speed at 12 m Max speed at 1200 m	20 kbs (1 kbs)	100 kbs 1 kbs	10 Mbs 100 kbs	35 Mbs 100 kbs
Max slew rate	30 V/µs	adjustable	n/a	n/a
Receiver input resistance	3..7 kΩ	≧ 4 kΩ	≧ 4 kΩ	≧ 12 kΩ
Driver load impedance	3..7 kΩ	≧ 450 Ω	100 Ω	54 Ω
Receiver input sensitivity	±3 V	±200 mV	±200 mV	±200 mV
Receiver input range	±15 V	±12 V	±10 V	-7..12 V
Max driver output voltage	±25 V	±6 V	±6 V	-7..12 V
Min driver output voltage (with load)	±5 V	±3.6 V	±2.0 V	±1.5 V

Equipment required

- Nu_LB-002 (Nuvoton learning board)

Reference:

1. Nu_LB-002 Rev 2.1 User's Manual
2. NuMicro™ NUC130_140 Technical Reference Manual EN V2.02
3. NuMicro™ NUC100 Series Driver Reference Guide V1.05.002

Procedure 1: UART program #1

1. Replace the content of the '**Smpl_Start_Kit.c**' with the 'UART1' lab file.
2. Compile the project, and run the program. (Add ScanKey.c to the project, from "C:\Nuvoton\BSP Library\NUC100SeriesBSP_CMSIS_v1.05.003\NuvotonPlatform_Keil\Src\NUC1xx-LB_002\ScanKey.c")
3. Study the program and work on assignments in the class.

```
27  #define DELAY300ms  300000 // The maximal delay time is 335000 us.
28
29  volatile uint8_t comRbuf[16];
30  volatile uint16_t comRbytes = 0;
31
32  char TEXT2[16] = "RX:             ";
33
34  //------------------------------------------------------------------UART_Callback
35  void UART0_INT_HANDLE(void) {
36    uint8_t bInChar[1] = {0xFF};
37
38    while (UART0->ISR.RDA_IF == 1) {  // Receive Data Available Interrupt Flag
39      DrvUART_Read(UART_PORT0,bInChar, 1);
40      if (comRbytes < 2) { // check if Buffer is full
41        comRbuf[comRbytes] = bInChar[0];
42        comRbytes++;
43        }
44      else if (comRbytes == 2) {
45        comRbuf[comRbytes] = bInChar[0];
46        comRbytes = 0;
47        sprintf(TEXT2+4, "%s", comRbuf);
48        print_lcd(2, TEXT2);
49        }
50      }
51    }
```

```
53  //------------------------------------------------------------------------MAIN
54  int32_t main() {
55    int8_t number;
56    uint8_t LCDcolumn = 1;
57    uint8_t dataout[1] = "1";
58
59    STR_UART_T sParam;
60
61    UNLOCKREG();
62    DrvSYS_Open(48000000);
63    LOCKREG();
64
65    Initial_pannel();  //call initial pannel function
66    clr_all_pannal();
67    print_lcd(0,"Smpl_UART0    ");
68
69    /* Set UART Pin */
70    DrvGPIO_InitFunction(E_FUNC_UART0);
71
72    /* UART Setting */
73    sParam.u32BaudRate = 9600;
74    sParam.u8cDataBits = DRVUART_DATABITS_8;
75    sParam.u8cStopBits = DRVUART_STOPBITS_1;
76    sParam.u8cParity = DRVUART_PARITY_NONE;
77    sParam.u8cRxTriggerLevel = DRVUART_FIFO_1BYTES;
78
79    /* Set UART Configuration */
80    if (DrvUART_Open(UART_PORT0,&sParam) != E_SUCCESS);
81
82    DrvUART_EnableInt(UART_PORT0, DRVUART_RDAINT, UART0_INT_HANDLE);
83
84    while (1) {
85      number = Scankey();
86
87      if (number == 1) {
88        dataout[0] = 0x31;
89        DrvUART_Write(UART_PORT0, dataout,1);
90
91        Show_Word(1,LCDcolumn,'1');
92        Show_Word(1,LCDcolumn+1,' ');
93        LCDcolumn++;
94          if (LCDcolumn > 14) LCDcolumn = 1;
95        DrvSYS_Delay(DELAY300ms);
96        }
97      if (number == 2) {
98        dataout[0] = 0x32;
99        DrvUART_Write(UART_PORT0, dataout,1);
100
101       Show_Word(1,LCDcolumn,'2');
102       Show_Word(1,LCDcolumn+1,' ');
103       LCDcolumn++;
104         if (LCDcolumn > 14) LCDcolumn = 1;
105       DrvSYS_Delay(DELAY300ms);
106       }
107     if (number == 3) {
108       dataout[0] = 0x33;
109       DrvUART_Write(UART_PORT0, dataout,1);
110
111       Show_Word(1,LCDcolumn,'3');
112       Show_Word(1,LCDcolumn+1,' ');
113       LCDcolumn++;
114         if (LCDcolumn > 14) LCDcolumn = 1;
115       DrvSYS_Delay(DELAY300ms);
116       }
117     if (number == 4) {
118       dataout[0] = 0x34;
119       DrvUART_Write(UART_PORT0, dataout,1);
120
121       Show_Word(1,LCDcolumn,'4');
122       Show_Word(1,LCDcolumn+1,' ');
123       LCDcolumn++;
124         if (LCDcolumn > 14) LCDcolumn = 1;
125       DrvSYS_Delay(DELAY300ms);
126       }
127     if (number == 5) {
128       dataout[0] = 0x35;
129       DrvUART_Write(UART_PORT0, dataout,1);
130
131       Show_Word(1,LCDcolumn,'5');
132       Show_Word(1,LCDcolumn+1,' ');
133       LCDcolumn++;
134         if (LCDcolumn > 14) LCDcolumn = 1;
135       DrvSYS_Delay(DELAY300ms);
136       }
137     }
138   //DrvUART_Close(UART_PORT0);
139   }
```

Procedure 2: UART program #2

1. Replace the content of the '**Smpl_Start_Kit.c**' with the 'UART2' lab file.
2. Compile the project, and run the program.
3. Study the program and work on assignments in the class.

```
27  volatile uint8_t comRbuf[16];
28  volatile uint16_t comRbytes = 0;
29
30  char TEXT1[16] = "TX: sending...  ";
31  char TEXT2[16] = "RX:             ";
32
33  //------------------------------------------------------------------UART_Callback
34 void UART_INT_HANDLE(void) {
35    uint8_t bInChar[1] = {0xFF};
36
37    while (UART0->ISR.RDA_IF == 1) {
38      DrvUART_Read(UART_PORT0, bInChar, 1);
39      if (comRbytes < 8) {  // check if Buffer is full
40        comRbuf[comRbytes] = bInChar[0];
41        comRbytes++;
42        }
43      else if (comRbytes == 8) {
44        comRbuf[comRbytes] = bInChar[0];
45        comRbytes = 0;
46        sprintf(TEXT2+4,"%s",comRbuf);
47        print_lcd(2,TEXT2);
48        }
49      }
50    }
51
52  //---------------------------------------------------------------------------MAIN
53 int32_t main() {
54    uint8_t  i = 0;
55    uint8_t  dataout[9] = "NuMicro0";
56
57    STR_UART_T sParam;
58
59    UNLOCKREG();
60    DrvSYS_Open(48000000);
61    LOCKREG();
62    Initial_pannel();  //call initial pannel function
63    clr_all_pannal();
64    print_lcd(0, "Smpl_UART0    ");
65
66    /* Set UART Pin */
67    DrvGPIO_InitFunction(E_FUNC_UART0);
68
69    /* UART Setting */
70    sParam.u32BaudRate = 9600;
71    sParam.u8cDataBits = DRVUART_DATABITS_8;
72    sParam.u8cStopBits = DRVUART_STOPBITS_1;
73    sParam.u8cParity = DRVUART_PARITY_NONE;
74    sParam.u8cRxTriggerLevel = DRVUART_FIFO_1BYTES;
75
76    /* Set UART Configuration */
77    if (DrvUART_Open(UART_PORT0,&sParam) != E_SUCCESS);
78
79    DrvUART_EnableInt(UART_PORT0, DRVUART_RDAINT, UART_INT_HANDLE);
80
81    while(1) {
82      dataout[7] = 0x30 + i;
83      DrvUART_Write(UART_PORT0, dataout, 8);
84      i++;
85      if (i >= 10) i = 0;
86      sprintf(TEXT1+14, "%d", i);
87      print_lcd(1, TEXT1);
88
89      DrvSYS_Delay(300000);
90      DrvSYS_Delay(300000);
91      DrvSYS_Delay(300000);
92    }
93    //DrvUART_Close(UART_PORT0);
94  }
```

Assignment(s)

Summarize what you suppose to learn in this class.