

LPC82X 培训资料

状态可配置定时器SCT 动手实验

MAY, 2016



EXTERNAL USE



SECURE CONNECTIONS
FOR A SMARTER WORLD

动手实验1

SCT产生PWM输出信号

内容

- 实验简介（目的，内容，结果）
- 软/硬件环境搭建
- 实验步骤

实验简介

- **目的**：通过本实验，理解和掌握LPC82x SCT实现PWM输出信号：
-设置输出不同周期的PWM信号
- **描述**：计数匹配产生2个事件，分别控制输出信号的高电平输出和低电平输出
- **结果**:LED灯闪烁

软/硬件环境搭建

- 硬件
 - 评估板：LPC824Lite-V1.0
- 工程位置
 - ..\peri_example\sctimer\sct_blinky\project_sct_blinky.uvprojx

实验步骤

- 第一步 – 更改计数器匹配条件，输出不同周期的PWM信号
- 第二步 – 根据连接指示，搭建好硬件环境
- 第三步 – 编译下载程序。运行，LED1灯闪烁，表明输出PWM信号

动手实验2

SCT实现计数器捕获

内容

- 实验简介（目的，内容，结果）
- 软/硬件环境搭建
- 实验步骤

实验简介

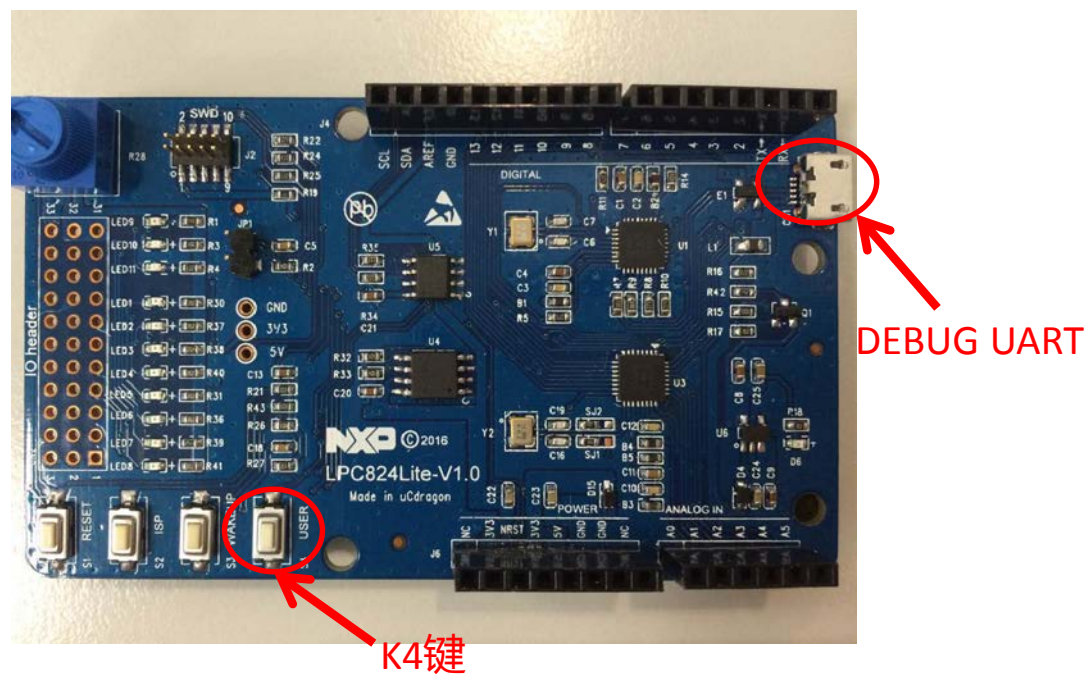
- **目的**：通过本实验，理解和掌握LPC82x SCT实现计数器捕获功能
- **描述**：利用P0_1做为SCT的SCT_INPUT0，按S4键产生下降沿的输入信号，该信号做为SCT事件0的触发信号捕获当前计数器值到捕获寄存器0，用DEBUG UART打印出捕获值
- **结果**：上位机串口打印出捕获计数器值

软/硬件环境搭建

- 硬件
 - 评估板：LPC824Lite-V1.0
- 工程位置
 - ..\peri_example\sctimer\sct_capture\project_sct_capture.uvprojx

硬件配置

- 无需特别的硬件配置
 - 使用DEBUG UART来输出调试字符串



实验步骤

- 第一步 – 根据连接指示，搭建好硬件环境
- 第二步 – 上位机设置串口助手为115200波特率，数据位8 - 校验0 - 停止位1
- 第三步 – 编译下载程序，运行
- 第四步 – 按下S4键，串口打印计数器捕获值

动手实验 3

SCT实现交通信号灯

内容

- 实验简介（目的，内容，结果）
- 软/硬件环境搭建
- 实验步骤

实验简介

- **目的**：通过本实验，理解和掌握LPC82x SCT硬件状态机实现交通信号灯
- **描述**：SCT实现4个状态分别代表红灯，绿灯，黄灯和黄灯闪烁，板子上的LED4，LED2和LED6分别表示红灯，黄灯和绿灯
- **结果**:LED模拟交通信号灯闪烁

软/硬件环境搭建

- 硬件
 - 评估板：LPC824Lite-V1.0
- 工程位置
 - ..\peri_example\sctimer\sct_tlight\project_sct_tlight.uvprojx

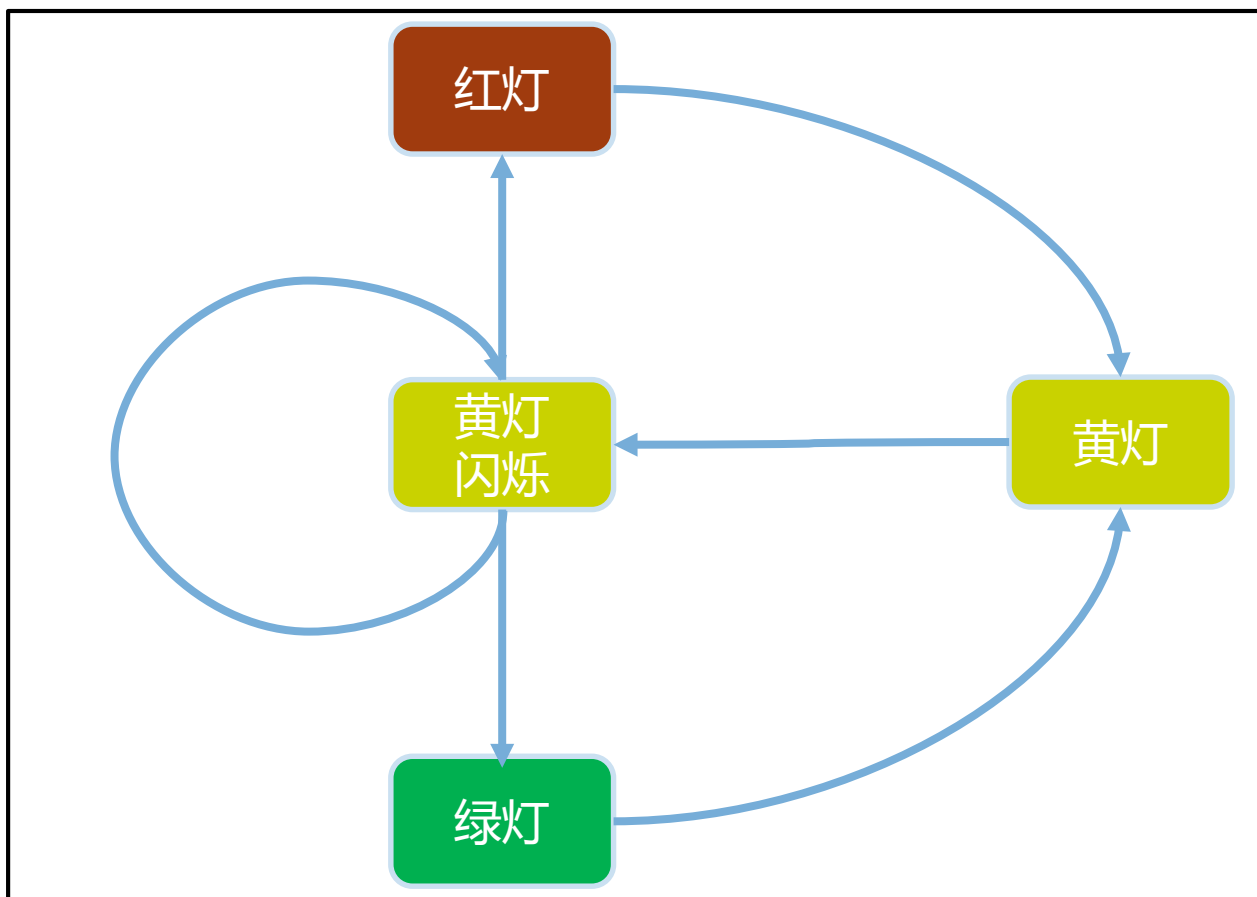
实验步骤

- 第一步 – 根据连接指示，搭建好硬件环境
- 第二步 – 编译下载程序，运行
- 第三步 – 盖板上三个LED2，4，6灯模拟交通灯交替闪烁

SCT状态机实例 - 交通灯

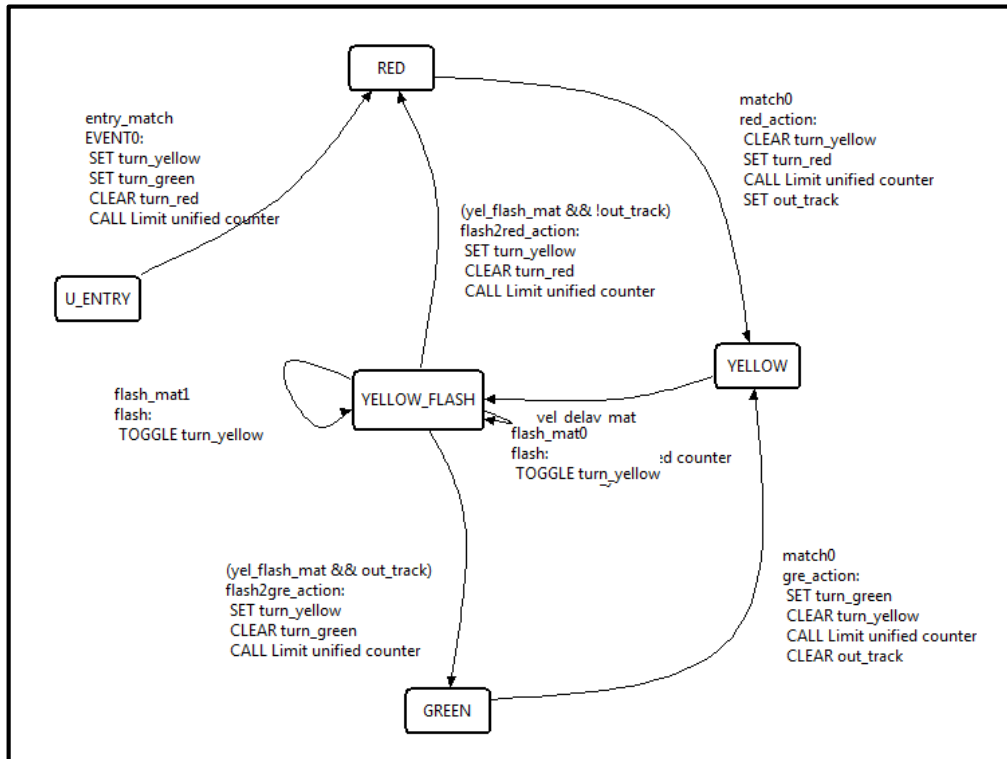
- 交通灯状态变换图

-红灯-> 黄灯-> 黄灯闪烁-> 绿灯-> 黄灯-> 黄灯闪烁-> 红灯



SCT状态机开发工具

- LPCXpresso IDE Red State
- Red State交通灯状态图及产生代码
- 启动后不需要任何软件干涉，解放CPU



```

void sct_fsm_init (void)
{
    LPC_SCT->MATCH[0].U = entry_mat;          /* entry_match */
    LPC_SCT->MATCHREL[0].U = entry_mat;
    LPC_SCT->MATCH[1].U = flash;              /* flash_mat0 */
    LPC_SCT->MATCHREL[1].U = flash;
    LPC_SCT->MATCH[2].U = flash1;             /* flash_mat1 */
    LPC_SCT->MATCHREL[2].U = flash1;
    LPC_SCT->MATCH[3].U = delay;              /* match0 */
    LPC_SCT->MATCHREL[3].U = delay;
    LPC_SCT->MATCH[4].U = yel_delay;          /* yel_delay_mat */
    LPC_SCT->MATCHREL[4].U = yel_delay;
    LPC_SCT->MATCH[5].U = yel_flash;         /* yel_flash_mat */
    LPC_SCT->MATCHREL[5].U = yel_flash;

    /* OUTPUT registers */
    LPC_SCT->OUT[5].SET = 0x00000002;         /* out_track */
    LPC_SCT->OUT[5].CLR = 0x00000008;
    LPC_SCT->OUT[2].SET = 0x00000009;         /* turn_green */
    LPC_SCT->OUT[2].CLR = 0x00000080;
    LPC_SCT->OUT[0].SET = 0x00000002;         /* turn_red */
    LPC_SCT->OUT[0].CLR = 0x00000041;
    LPC_SCT->OUT[1].SET = 0x000000F5;         /* turn_yellow */
    LPC_SCT->OUT[1].CLR = 0x0000003A;

    /* Conflict resolution register */
    LPC_SCT->RES = (LPC_SCT->RES & ~0x0000000C) | 0x0000000C;

    /* EVENT registers */
    LPC_SCT->EV[0].CTRL = 0x0000D000;        /* U: --> state RED */
    LPC_SCT->EV[0].STATE = 0x00000001;
    LPC_SCT->EV[1].CTRL = 0x00015003;        /* U: --> state YELLOW */
    LPC_SCT->EV[1].STATE = 0x00000002;
    LPC_SCT->EV[2].CTRL = 0x00025004;        /* U: --> state YELLOW_FLASH */
    LPC_SCT->EV[2].STATE = 0x00000004;
    LPC_SCT->EV[3].CTRL = 0x00015003;        /* U: --> state YELLOW */
    LPC_SCT->EV[3].STATE = 0x00000008;
    LPC_SCT->EV[6].CTRL = 0x0000F165;        /* U: --> state RED */
    LPC_SCT->EV[6].STATE = 0x00000010;
    LPC_SCT->EV[7].CTRL = 0x0001FD65;        /* U: --> state GREEN */
    LPC_SCT->EV[7].STATE = 0x00000010;
    LPC_SCT->EV[4].CTRL = 0x00025001;        /* U: --> state YELLOW_FLASH */
    LPC_SCT->EV[4].STATE = 0x00000010;
    LPC_SCT->EV[5].CTRL = 0x00025002;        /* U: --> state YELLOW_FLASH */
    LPC_SCT->EV[5].STATE = 0x00000010;

    /* STATE registers */
    LPC_SCT->STATE_L = 0;
    LPC_SCT->LIMIT_L = 0x000000CF;
}
    
```



普通Timer实现交通灯对比

- 状态机需要软件实现
- 高度CPU消耗

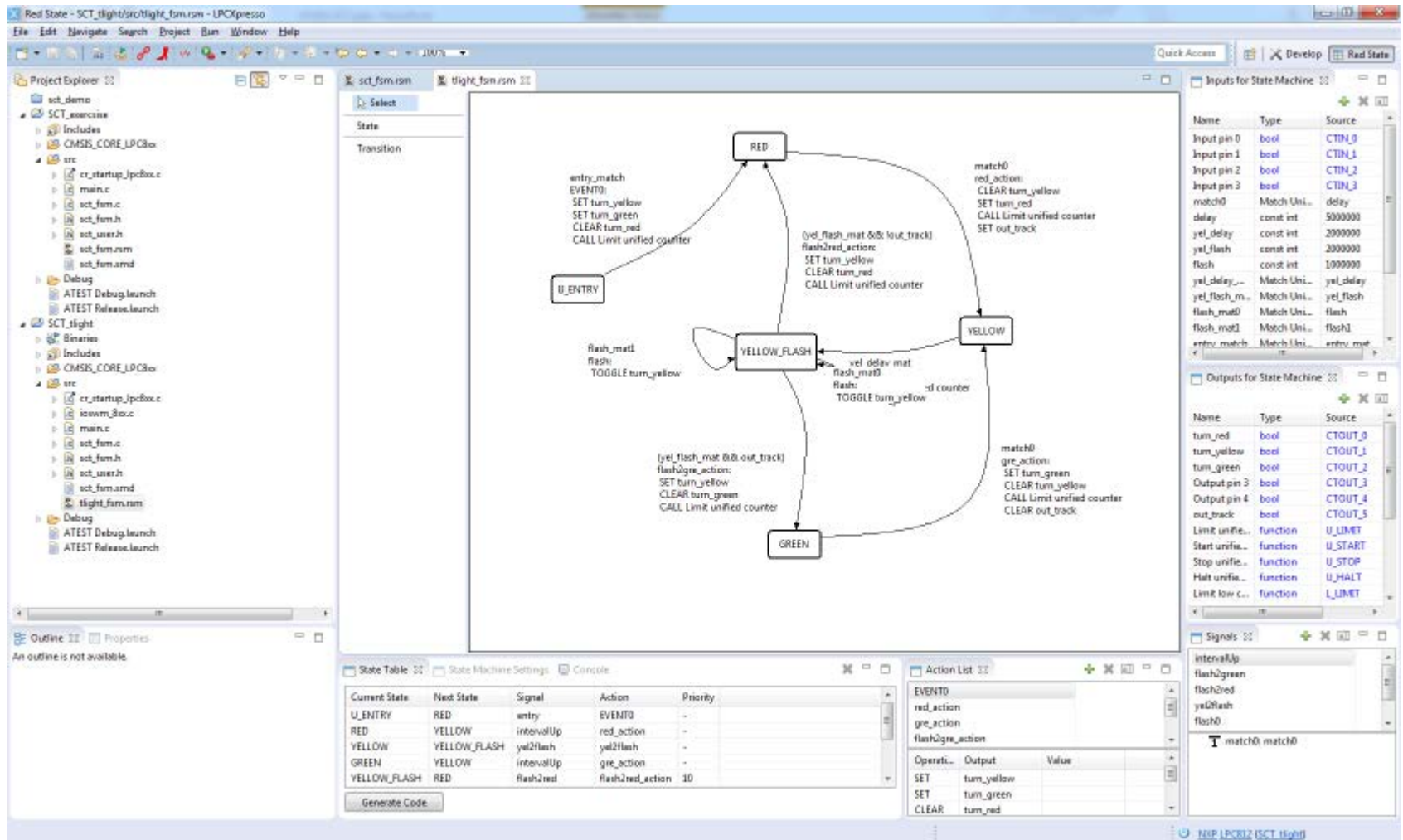
```
init()
{
    red_led = out;
    yellow_led = out;
    green_led = out;

    match = red_interval;
    match_interrupt = enable;
    state = red;
    red_led = up;
    counter = start;
}
```

```
match_state_isr()
{
    switch(state)
    {
        case red:
            red_led = out;
            yellow_led = up;
            red_flag = 1;
            match = yellow_interval;
            state = yellow;
            counter = start;
        case yellow:
            yellow_led = flash;
            match = yellowFlash_interval;
            state = yellow_flash;
            counter = start;

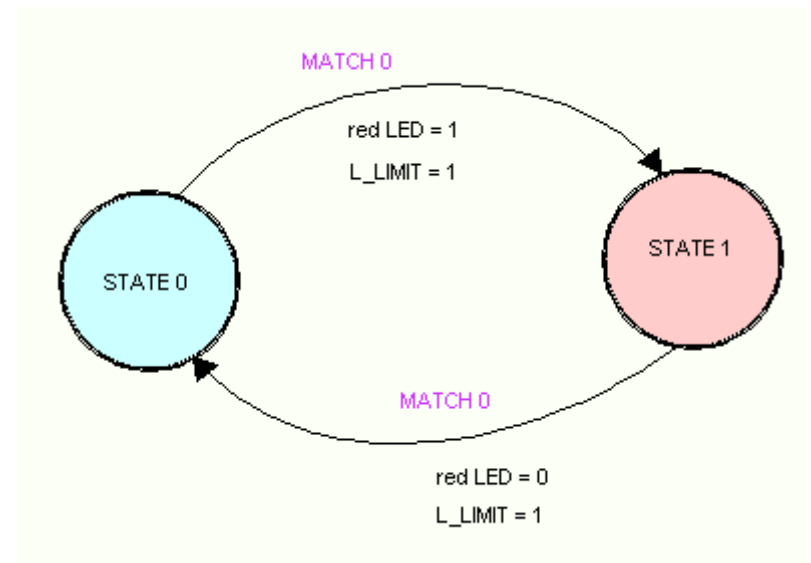
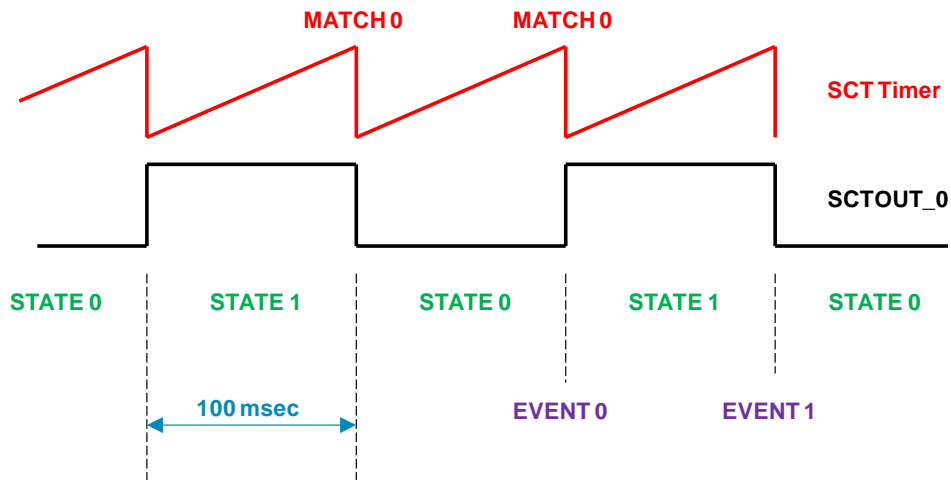
        case yellow_flash:
            yellow_led = out;
            if(red_flag == 1)
            {
                green_led = up;
                state = green;
                match = green_interval;
                counter = start;
            }
            else
            {
                red_led = up;
                state = red;
                match = red_interval;
                counter = start;
            }
        case green:
            green_led = out;
            yellow_led = up;
            red_flag = 0;
            match_interval = yellow_interval;
            state = yellow;
            counter = start;
        }
    }
```

Red State介绍 – 图形化的SCT配置工具



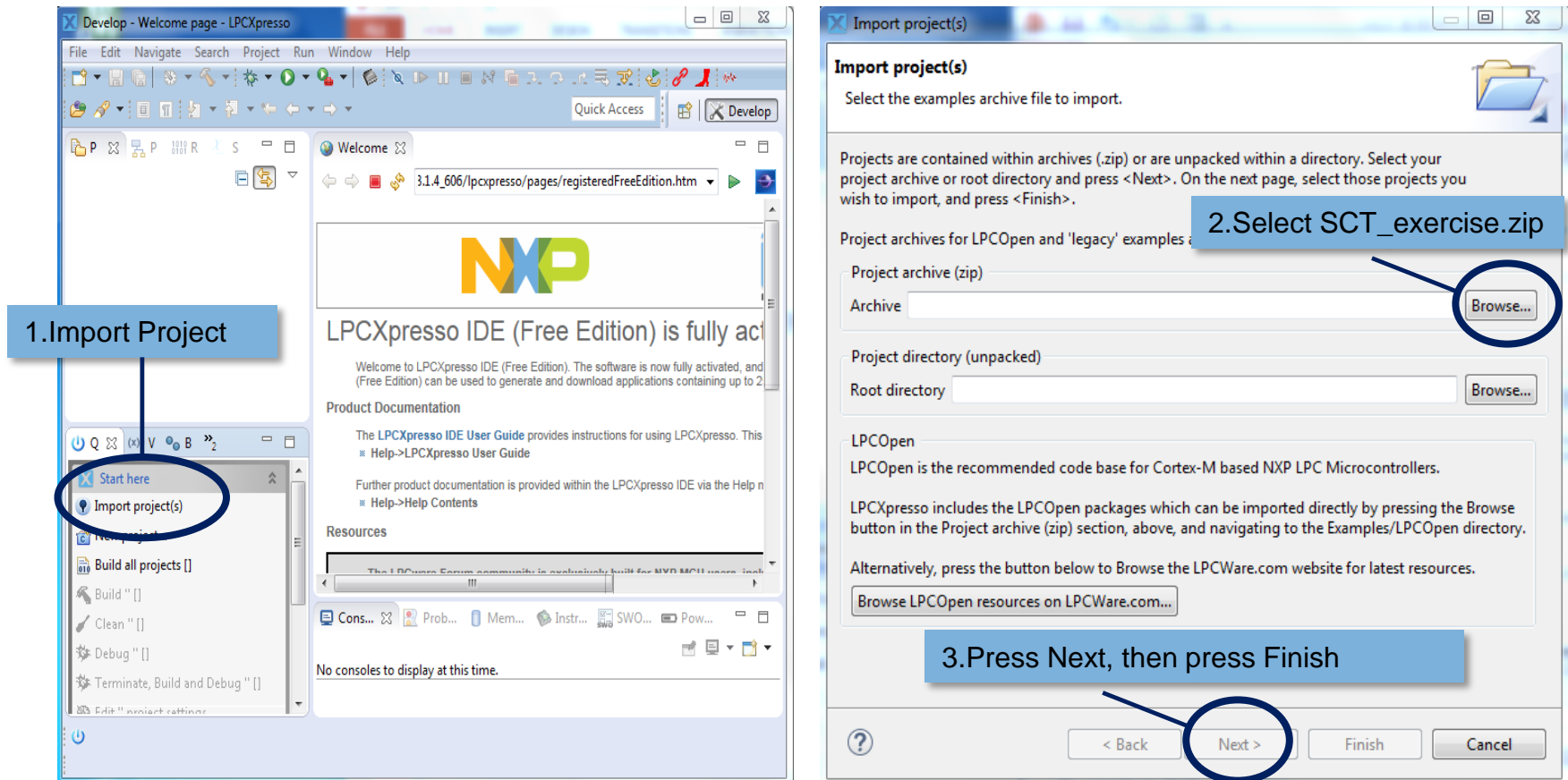
Red State实验

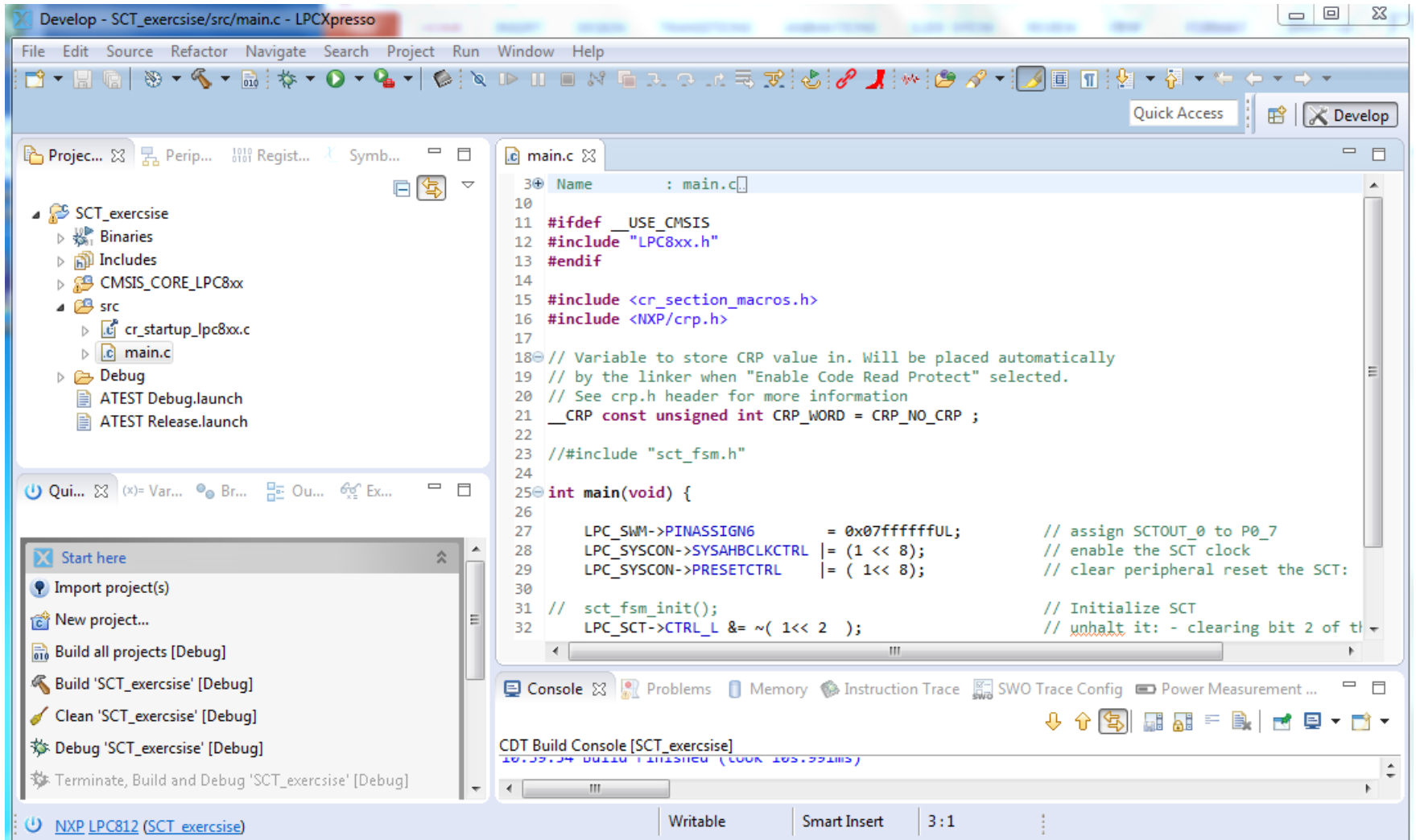
- 使用 SCT 和 Red State 来做前述的实验
- 连接 RED led (P0_17) 到 SCTOUT_0



步骤 - 1

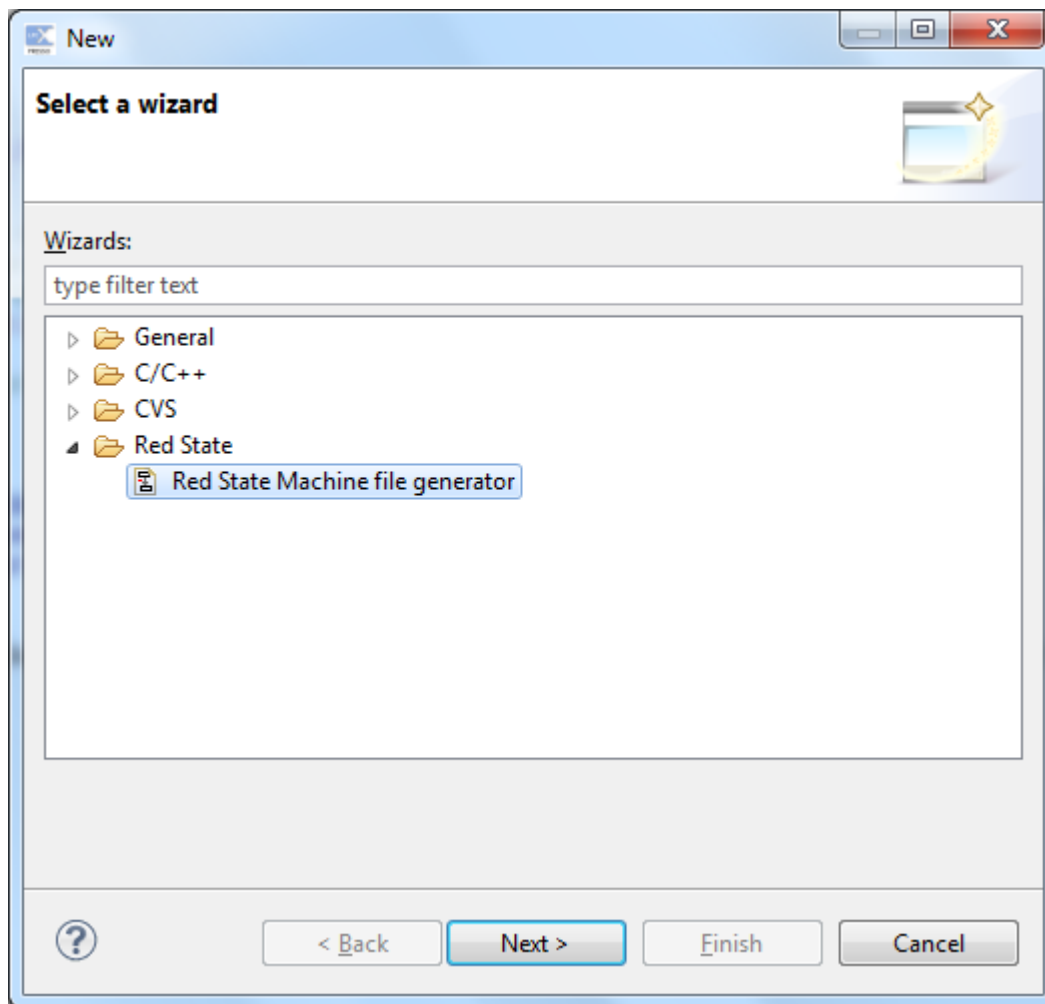
- 打开 LPCXpresso , 打开工程 SCT_exercise
-..\peri_example\sctimer\SCT_exercise.zip





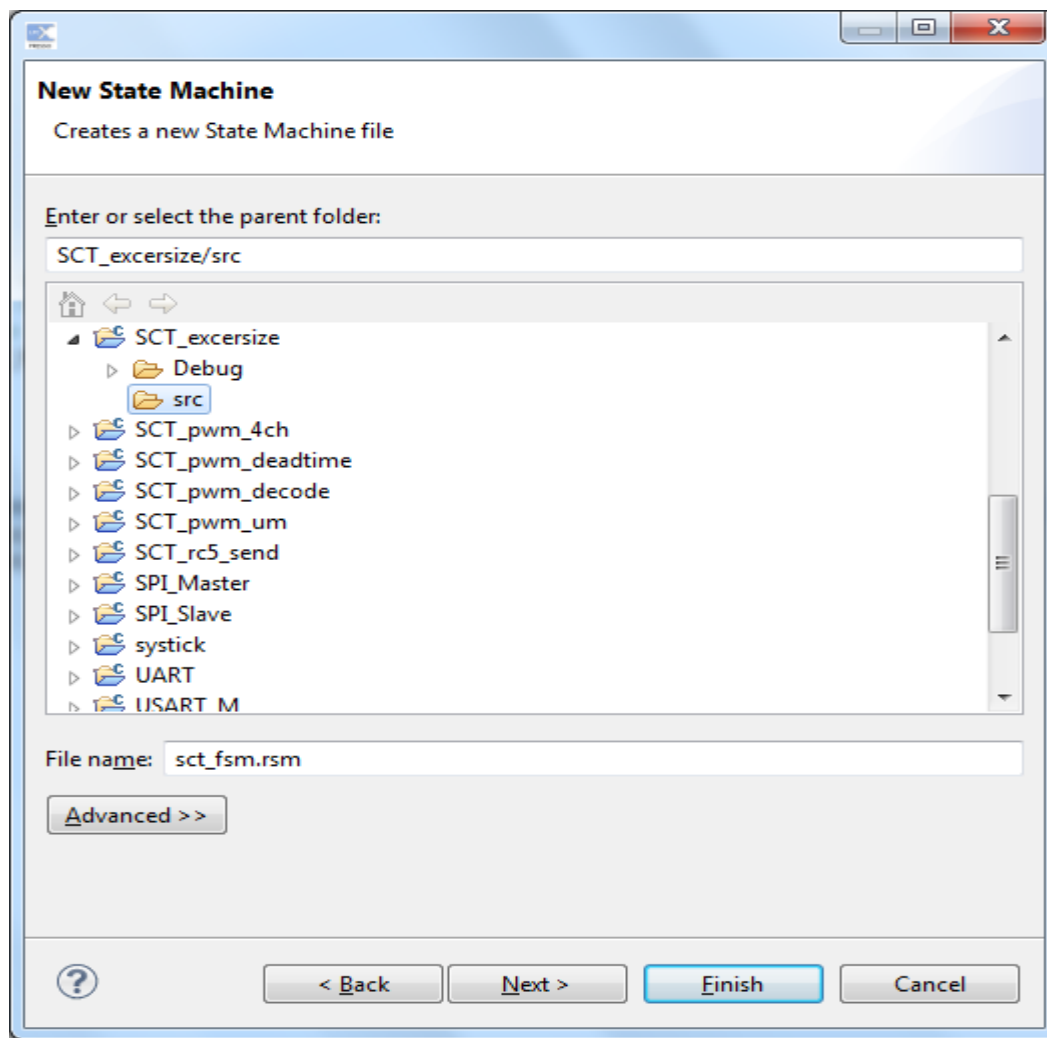
步骤 - 2

- 选中 file/new/other
- 选中 Red State . . .



步骤 - 3

- 选中上一级目录
- 输入文件名
- 点击 Next
- 点击 Finish



Red State - SCT_excercise/src/sct_fsm.rsm - LPCXpresso

File Edit Navigate Search Project Run Window Help

Quick Access Develop Red State

Project Explorer

- MRT_blinky
- PMU
- SCT_blinky
- SCT_excercise
 - Binaries
 - Includes
 - src
 - cr_startup_lpc8xx.c
 - main.c
 - sct_fsm.rsm
 - Debug
 - ATEST Debug.launch
 - ATEST Release.launch
- SCT_pwm_4ch
- SCT_pwm_deadtime
- SCT_pwm_decode
- SCT_pwm_um

main.c **sct_fsm.rsm**

Select

State

Transition

U_ALWAYS

U_ENTRY

Inputs for State Machine

Name	Type	Source
Input pin 0	bool	CTIN_0
Input pin 1	bool	CTIN_1
Input pin 2	bool	CTIN_2
Input pin 3	bool	CTIN_3

Outputs for State Machine

Name	Type	Source	preload
Output pin 0	bool	CTOUT_0	
Output pin 1	bool	CTOUT_1	
Output pin 2	bool	CTOUT_2	
Output pin 3	bool	CTOUT_3	
Limit unifie...	function	U_LIMIT	
Start unifie...	function	U_START	
Stop unifie...	function	U_STOP	
Halt unifie...	function	U_HALT	

State Table **State Machine Settings**

Current State	Next State	Signal	Action	Priori
(click to add)	(click to add)	(click to add)	(click to add)	-

Generate Code

Action List

Operati...	Output	Value

Signals **Console**

CDT Build Console [SCT_excercise]

```

C:\LPC811\bin\lpc811\lpc811.exe -p -t -b -d -e -f -g -h -i -j -k -l -m -n -o -p -q -r -s -t -u -v -w -x -y -z -A -B -C -D -E -F -G -H -I -J -K -L -M -N -O -P -Q -R -S -T -U -V -W -X -Y -Z -AA -AB -AC -AD -AE -AF -AG -AH -AI -AJ -AK -AL -AM -AN -AO -AP -AQ -AR -AS -AT -AU -AV -AW -AX -AY -AZ -BA -BB -BC -BD -BE -BF -BG -BH -BI -BJ -BK -BL -BM -BN -BO -BP -BQ -BR -BS -BT -BU -BV -BW -BX -BY -BZ -CA -CB -CC -CD -CE -CF -CG -CH -CI -CJ -CK -CL -CM -CN -CO -CP -CQ -CR -CS -CT -CU -CV -CW -CX -CY -CZ -DA -DB -DC -DD -DE -DF -DG -DH -DI -DJ -DK -DL -DM -DN -DO -DP -DQ -DR -DS -DT -DU -DV -DW -DX -DY -DZ -EA -EB -EC -ED -EE -EF -EG -EH -EI -EJ -EK -EL -EM -EN -EO -EP -EQ -ER -ES -ET -EU -EV -EW -EX -EY -EZ -FA -FB -FC -FD -FE -FF -FG -FH -FI -FJ -FK -FL -FM -FN -FO -FP -FQ -FR -FS -FT -FU -FV -FW -FX -FY -FZ -GA -GB -GC -GD -GE -GF -GG -GH -GI -GJ -GK -GL -GM -GN -GO -GP -GQ -GR -GS -GT -GU -GV -GW -GX -GY -GZ -HA -HB -HC -HD -HE -HF -HG -HH -HI -HJ -HK -HL -HM -HN -HO -HP -HQ -HR -HS -HT -HU -HV -HW -HX -HY -HZ -IA -IB -IC -ID -IE -IF -IG -IH -II -IJ -IK -IL -IM -IN -IO -IP -IQ -IR -IS -IT -IU -IV -IW -IX -IY -IZ -JA -JB -JC -JD -JE -JF -JG -JH -JI -JJ -JK -JL -JM -JN -JO -JP -JQ -JR -JS -JT -JU -JV -JW -JX -JY -JZ -KA -KB -KC -KD -KE -KF -KG -KH -KI -KJ -KK -KL -KM -KN -KO -KP -KQ -KR -KS -KT -KU -KV -KW -KX -KY -KZ -LA -LB -LC -LD -LE -LF -LG -LH -LI -LJ -LK -LL -LM -LN -LO -LP -LQ -LR -LS -LT -LU -LV -LW -LX -LY -LZ -MA -MB -MC -MD -ME -MF -MG -MH -MI -MJ -MK -ML -MM -MN -MO -MP -MQ -MR -MS -MT -MU -MV -MW -MX -MY -MZ -NA -NB -NC -ND -NE -NF -NG -NH -NI -NJ -NK -NL -NM -NN -NO -NP -NQ -NR -NS -NT -NU -NV -NW -NX -NY -NZ -OA -OB -OC -OD -OE -OF -OG -OH -OI -OJ -OK -OL -OM -ON -OO -OP -OQ -OR -OS -OT -OU -OV -OW -OX -OY -OZ -PA -PB -PC -PD -PE -PF -PG -PH -PI -PJ -PK -PL -PM -PN -PO -PP -PQ -PR -PS -PT -PU -PV -PW -PX -PY -PZ -QA -QB -QC -QD -QE -QF -QG -QH -QI -QJ -QK -QL -QM -QN -QO -QP -QQ -QR -QS -QT -QU -QV -QW -QX -QY -QZ -RA -RB -RC -RD -RE -RF -RG -RH -RI -RJ -RK -RL -RM -RN -RO -RP -RQ -RR -RS -RT -RU -RV -RW -RX -RY -RZ -SA -SB -SC -SD -SE -SF -SG -SH -SI -SJ -SK -SL -SM -SN -SO -SP -SQ -SR -SS -ST -SU -SV -SW -SX -SY -SZ -TA -TB -TC -TD -TE -TF -TG -TH -TI -TJ -TK -TL -TM -TN -TO -TP -TQ -TR -TS -TT -TU -TV -TW -TX -TY -TZ -UA -UB -UC -UD -UE -UF -UG -UH -UI -UJ -UK -UL -UM -UN -UO -UP -UQ -UR -US -UT -UU -UV -UW -UX -UY -UZ -VA -VB -VC -VD -VE -VF -VG -VH -VI -VJ -VK -VL -VM -VN -VO -VP -VQ -VR -VS -VT -VU -VV -VW -VX -VY -VZ -WA -WB -WC -WD -WE -WF -WG -WH -WI -WJ -WK -WL -WM -WN -WO -WP -WQ -WR -WS -WT -WU -WV -WW -WX -WY -WZ -XA -XB -XC -XD -XE -XF -XG -XH -XI -XJ -XK -XL -XM -XN -XO -XP -XQ -XR -XS -XT -XU -XV -XW -XX -XY -XZ -YA -YB -YC -YD -YE -YF -YG -YH -YI -YJ -YK -YL -YM -YN -YO -YP -YQ -YR -YS -YT -YU -YV -YW -YX -YY -YZ -ZA -ZB -ZC -ZD -ZE -ZF -ZG -ZH -ZI -ZJ -ZK -ZL -ZM -ZN -ZO -ZP -ZQ -ZR -ZS -ZT -ZU -ZV -ZW -ZX -ZY -ZZ
  
```

hex filename

hex	data	bss	dec
516	0	0	516
204	SCT_excercise.axf		

An outline is not available.

步骤 - 4

- Delete U_ALWAYS
- Enter 2 state transitions
- Enter Signal names

Current State	Next State	Signal	Action	Priori
U_ENTRY	LED_ON	MATCH_0	(click to add)	-
LED_ON	U_ENTRY	MATCH_0	(click to add)	-
(click to add)	(click to add)	(click to add)	(click to add)	-

Operati...	Output	Value

步骤 - 5

The screenshot shows the NXP SCT (State Configuration Tool) interface. The main window displays a state transition diagram with two states, U_ENTRY and LED_ON, connected by a transition labeled MATCH_0:(no test) (no action). The 'Inputs for State Machine' table is highlighted with a blue oval, and a blue callout box points to the '+' icon in the top right corner of the table, indicating the 'ADD' button. The 'Outputs for State Machine' table is also visible. The bottom of the interface shows the 'Signal Settings' table, the 'Action List' table, and the 'Console' window.

Name	Type	Source
Input pin 0	bool	CTIN_0
Input pin 1	bool	CTIN_1
Input pin 2	bool	CTIN_2
Input pin 3	bool	CTIN_3
delay	const int	10000000
match0	Match Uni...	delay

Name	Type	Source	preload
Output pin 0	bool	CTOUT_0	
Output pin 1	bool	CTOUT_1	
Output pin 2	bool	CTOUT_2	
Output pin 3	bool	CTOUT_3	
Limit unifie...	function	U_LIMIT	
Start unifie...	function	U_START	
Stop unifie...	function	U_STOP	
Halt unifie...	function	U_HALT	

Signal	Action	Priori
MATCH_0	(click to add)	-
MATCH_0	(click to add)	-
)	(click to add)	-

Operati...	Output	Value

- Select ADD (+) and
- Enter delay value
- Enter Input match 0

步骤 - 6

The screenshot displays the NXP IDE State Machine Editor interface. At the top, a state transition diagram shows two states: U_ENTRY and LED_ON. Transitions are labeled with red text: MATCH_0:(no test) EVENT 1: CLEAR Output pin 0 CALL Limit unified counter (from U_ENTRY to LED_ON) and MATCH_0:(no test) EVENT 0: SET Output pin 0 CALL Limit unified counter (from LED_ON to U_ENTRY). Below the diagram, the 'State Table' panel shows the following data:

Current State	Next State	Signal	Action
U_ENTRY	LED_ON	MATCH_0	EVENT 0
LED_ON	U_ENTRY	MATCH_0	EVENT 1
(click to add)	(click to add)	(click to add)	(click to add)

The 'Action List' panel shows two events defined:

- EVENT 0: SET Output pin 0, CALL Limit unified c...
- EVENT 1: CLEAR Output pin 0, CALL Limit unified c...

On the right, the 'Name' and 'Type' tables are visible:

Name	Type	Source
Input pin 0	bool	CTIN
Input pin 1	bool	CTIN
Input pin 2	bool	CTIN
Input pin 3	bool	CTIN
delay	const int	10000
match0	Match Uni...	delay

Below this, the 'Outputs for State Machine' table is shown:

Name	Type	Source
Output pin 0	bool	CTOL
Output pin 1	bool	CTOL
Output pin 2	bool	CTOL
Output pin 3	bool	CTOL
Limit unifie...	function	U_LIM
Start unifie...	function	U_ST
Stop unifie...	function	U_ST
Halt unifie...	function	U_HA

At the bottom right, the 'Console' panel shows the command: CDT Build Console [SCT_blinky].

Add actions to the state transitions

- Add 2 Actions (EVENTS) and for
- EV0 operations SET and LIMIT
- EV1 operations CLEAR and LIMIT

步骤 - 7

The screenshot displays the State Machine Editor interface with the following components:

- State Transition Diagram:** A diagram showing two states, `U_ENTRY` and `LED_ON`, connected by two transitions. The top transition is labeled `match0` and `EVENT 1: CLEAR Output 0 CALL Limit unified counter`. The bottom transition is labeled `match0` and `EVENT 0: SET Output 0 CALL Limit unified counter`.
- Inputs for State Machine:** A table listing inputs and their sources.

Name	Type	Source
Input pin 0	bool	CTIN_0
Input pin 1	bool	CTIN_1
Input pin 2	bool	CTIN_2
Input pin 3	bool	CTIN_3
delay	const int	10000000
match0	Match Uni...	delay
- Outputs for State Machine:** A table listing outputs and their sources.

Name	Type	Source	preload
Output 0	bool	CTOUT_0	TRUE
Output pin 1	bool	CTOUT_1	
Output pin 2	bool	CTOUT_2	
Output pin 3	bool	CTOUT_3	
Limit unifie...	function	U_LIMIT	
Start unifie...	function	U_START	
Stop unifie...	function	U_STOP	
Halt unifie...	function	U_HALT	
- State Table:** A table showing the current state, next state, signal, and action.

Current State	Next State	Signal	Action	Priori
U_ENTRY	LED_ON	MATCH_0	EVENT 0	-
LED_ON	U_ENTRY	MATCH_0	EVENT 1	-
(click to add)	(click to add)	(click to add)	(click to add)	-
- Action List:** A table showing the operations, outputs, and values for each event.

Operati...	Output	Value
CLEAR	Output 0	
CALL	Limit unified c...	
- Signals:** A panel showing the signal `MATCH_0` with a value of `1` and the text `match0: match0`.

Add signal

步骤 - 8

The screenshot displays the State Machine Editor interface with the following components:

- State Transition Diagram:** Shows two states, `U_ENTRY` and `LED_ON`, connected by two transitions.
 - match0 EVENT 1:** CLEAR Output 0, CALL Limit unified counter (from `U_ENTRY` to `LED_ON`).
 - match0 EVENT 0:** SET Output 0, CALL Limit unified counter (from `LED_ON` to `U_ENTRY`).
- State Table:**

Current State	Next State	Signal	Action	Priori
U_ENTRY	LED_ON	MATCH_0	EVENT 0	-
LED_ON	U_ENTRY	MATCH_0	EVENT 1	-
(click to add)	(click to add)	(click to add)	(click to add)	-
- Inputs for State Machine:**

Name	Type	Source
Input pin 0	bool	CTIN_0
Input pin 1	bool	CTIN_1
Input pin 2	bool	CTIN_2
Input pin 3	bool	CTIN_3
delay	const int	10000000
match0	Match Uni...	delay
- Outputs for State Machine:** (Highlighted with a blue circle)

Name	Type	Source	preload
Output 0	bool	CTOUT_0	TRUE
Output pin 1	bool	CTOUT_1	
Output pin 2	bool	CTOUT_2	
Output pin 3	bool	CTOUT_3	
Limit unifie...	function	U_LIMIT	
Start unifie...	function	U_START	
Stop unifie...	function	U_STOP	
Halt unifie...	function	U_HALT	
- Action List:**

Operati...	Output	Value
CLEAR	Output 0	
CALL	Limit unified c...	
- Signals:**

Signal
MATCH_0

Enter preset value for output 0

步骤 - 9

The screenshot shows the Red State IDE interface. The main window displays a state machine diagram for 'sct_fsm.rsm'. A dialog box titled 'Generating code' is open, showing the following message:

Successfully generated code in:
 C:\localdata\LPC800_LPCpresso\SCT_exercise\src\sct_fsm.smd
 C:\localdata\LPC800_LPCpresso\SCT_exercise\src\sct_user.h

The State Table panel at the bottom left contains the following data:

Current State	Next State	Signal	Action	Priori
U_ENTRY	LED_ON	MATCH_0	EVENT 0	-
LED_ON	U_ENTRY	MATCH_0	EVENT 1	-
(click to add)	(click to add)	(click to add)	(click to add)	-

The Action List panel at the bottom center shows the following actions:

Operati...	Output	Value
CLEAR	Output 0	
CALL	Limit unified c...	

The Signals panel at the bottom right shows the following signal:

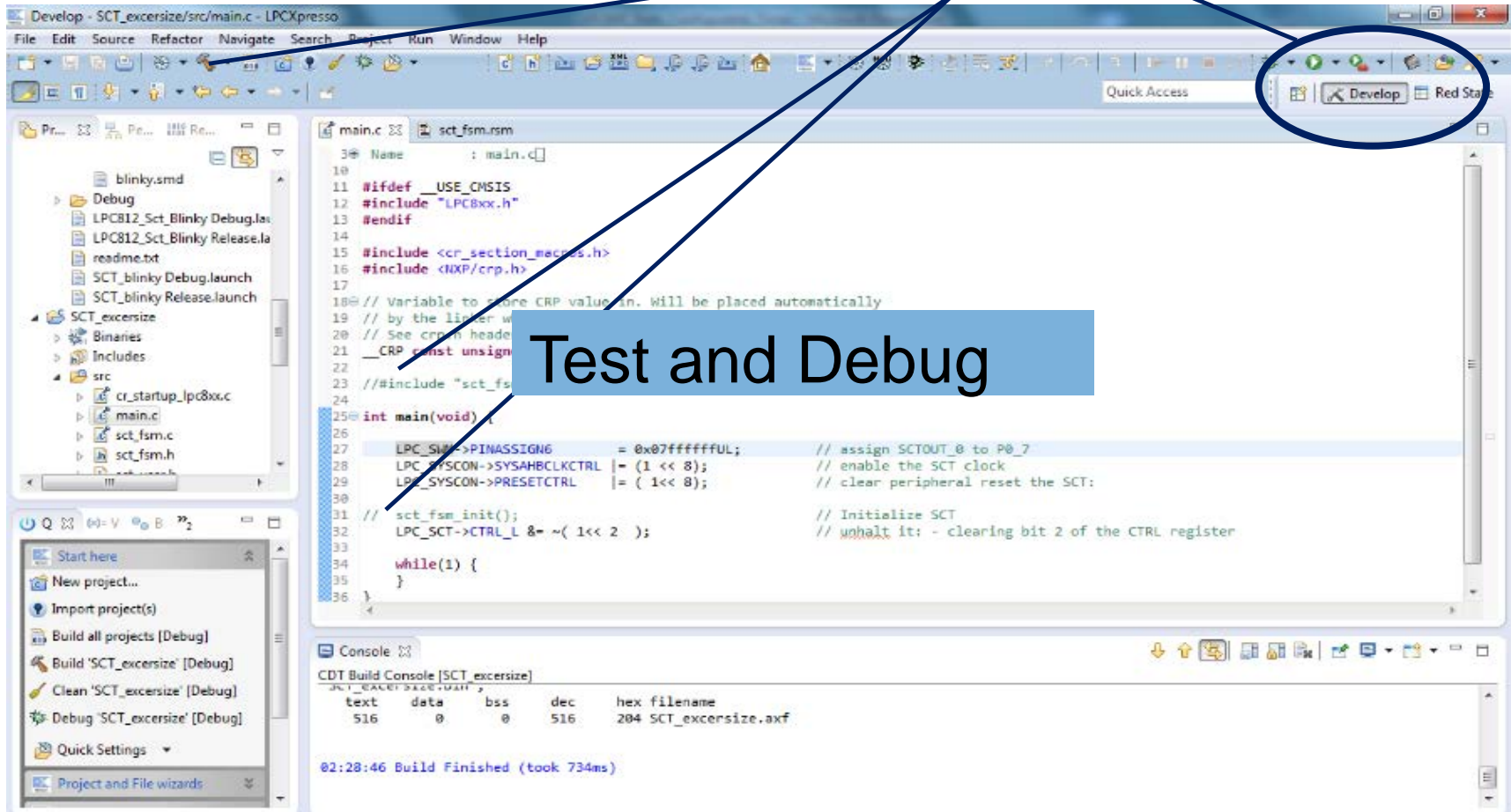
Signal
MATCH_0

Press GENERATE CODE

步骤 - 10

Leave RedState
And build project
Check

Test and Debug





SECURE CONNECTIONS
FOR A SMARTER WORLD