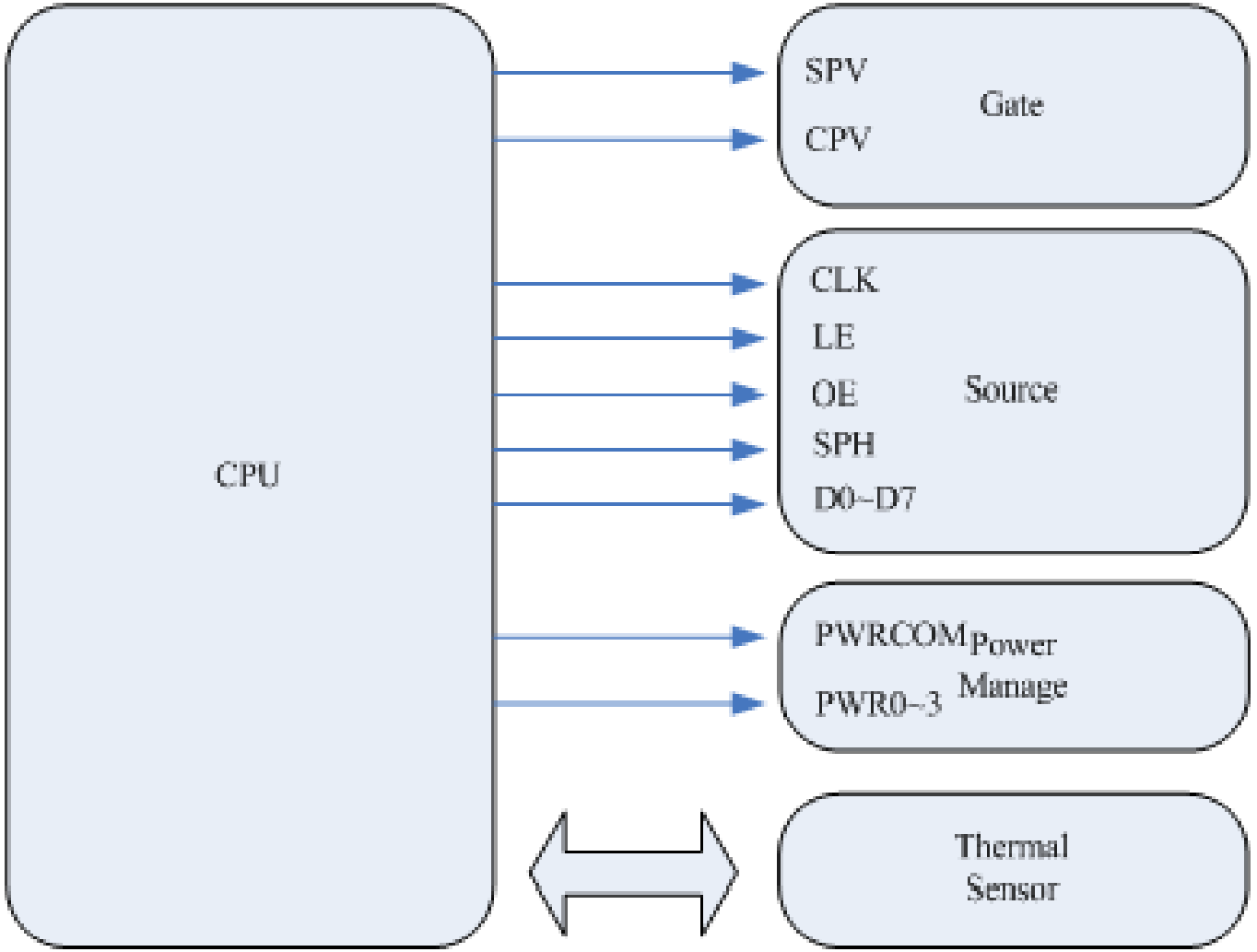
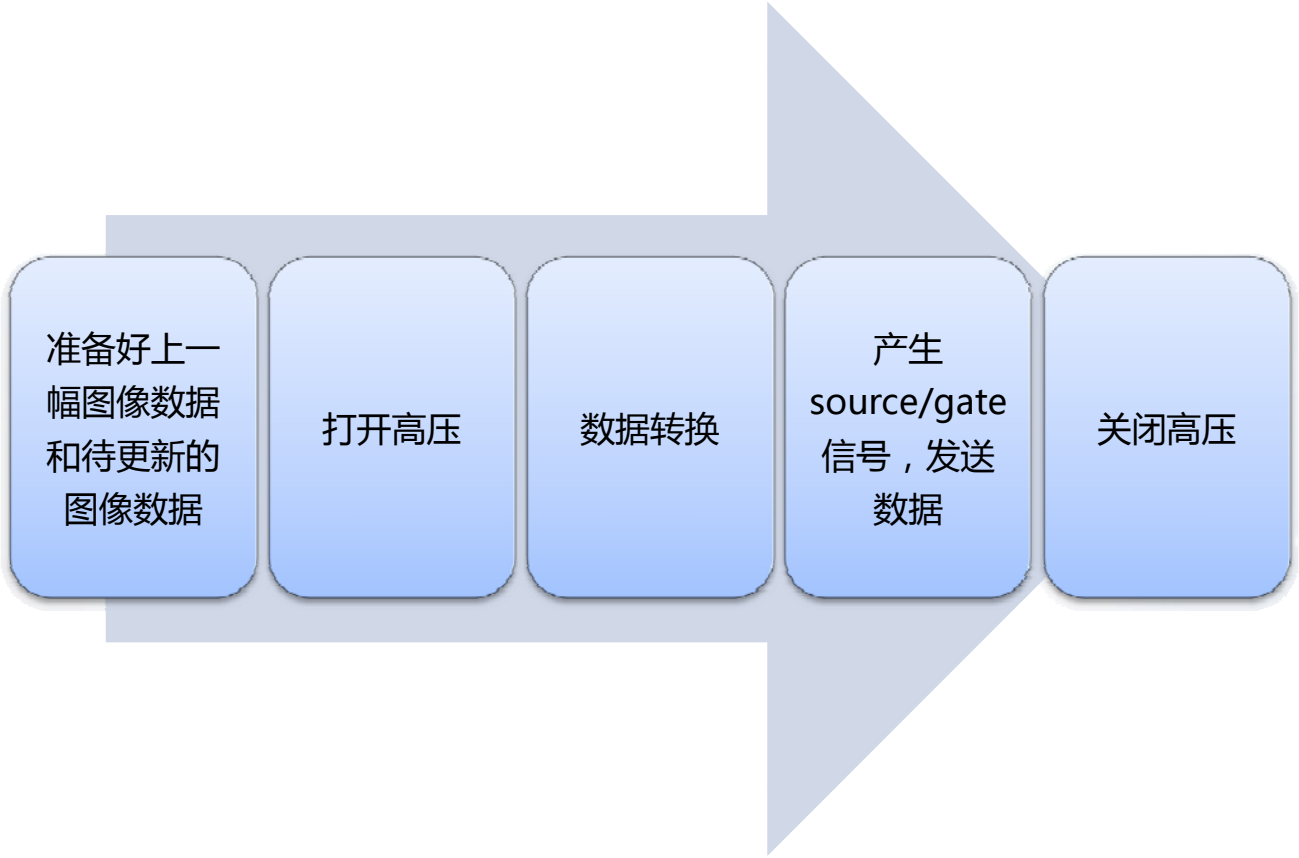
MCU模拟Tcon直接驱动EPD

2012.06.20

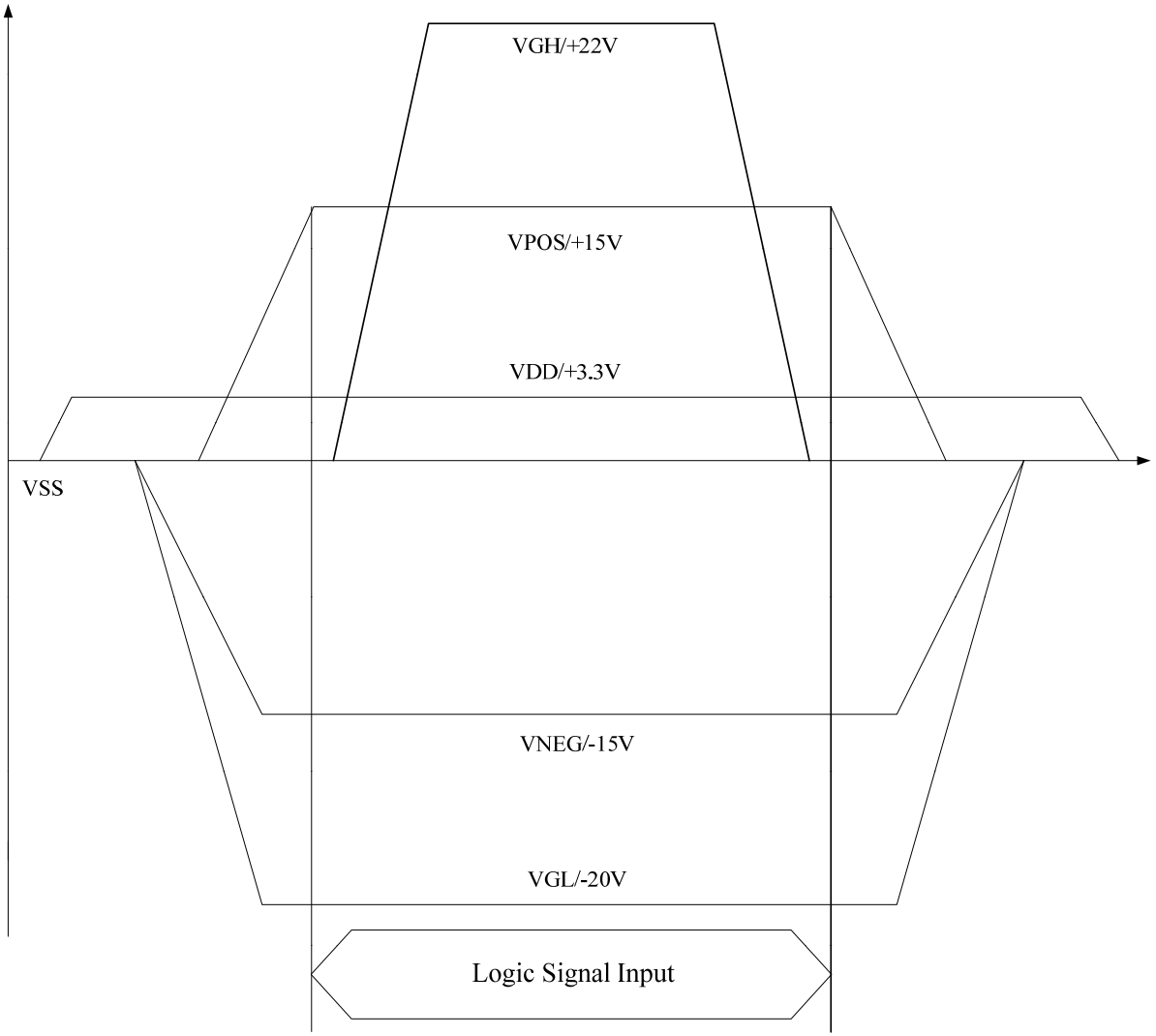
CPU与EPD的连接框图



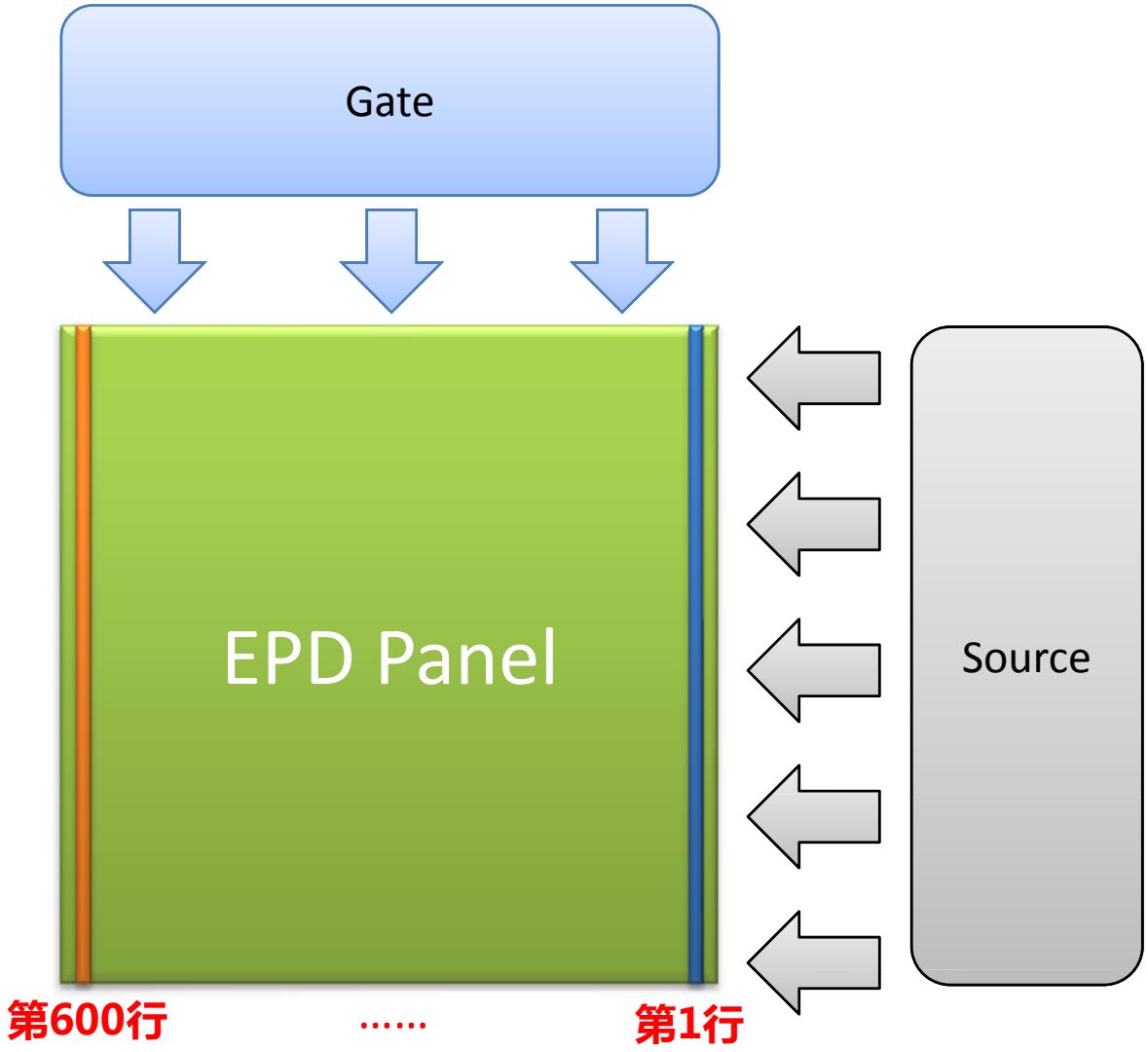
EPD驱动流程



Power时序

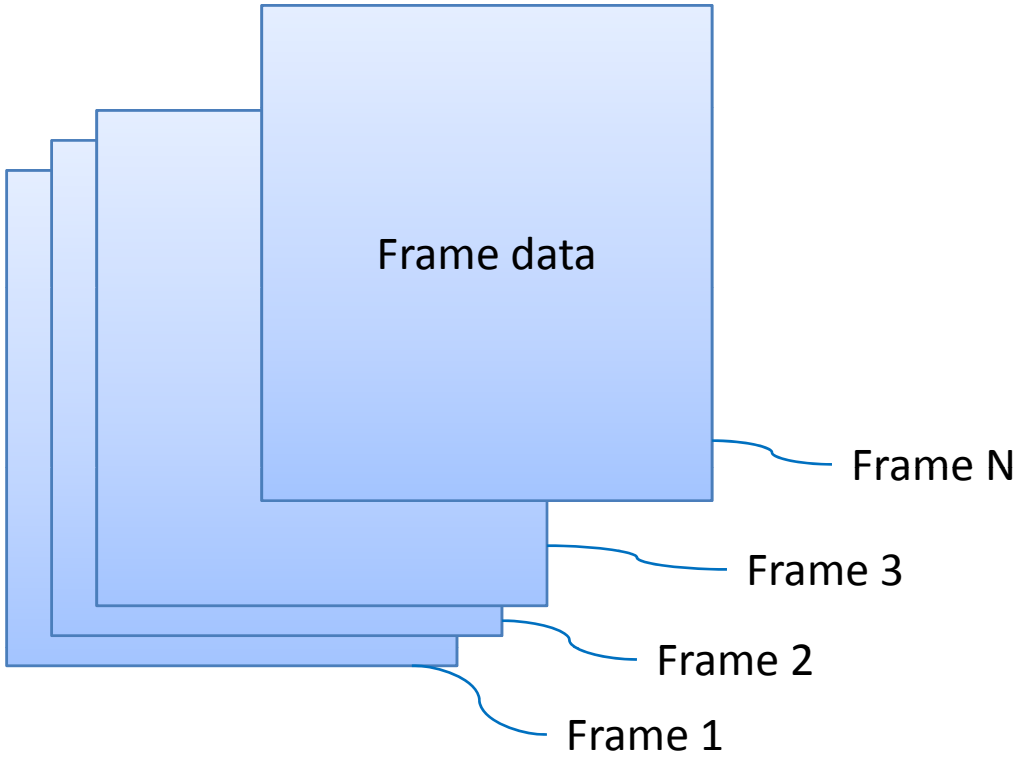

VGH/+22V
VPOS/+15V
VDD/+3.3V
VSS
VNEG/-15V
VGL/-20V
Logic Signal Input

Source/gate驱动方式（一）



Source/gate驱动方式（二）

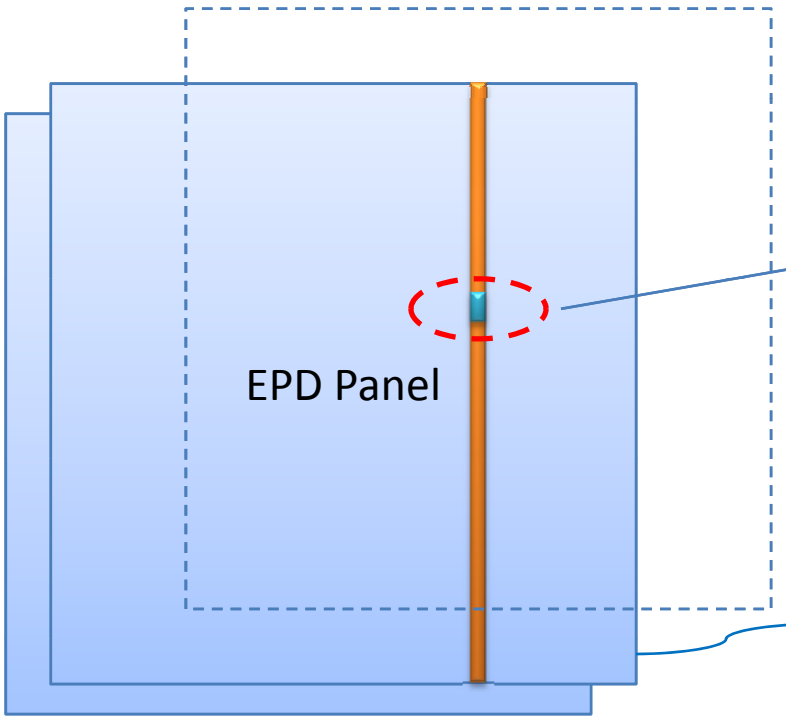
- EPD像素由source芯片进行驱动，每次source芯片把一行的数据准备好后，gate芯片就打开1行进行驱动
- Gate芯片逐行打开，600行全扫描完为1个frame
- 由于EPD响应速度不快，故不能像普通的TFT屏只驱动一帧即可显示，而是需要多个frame叠加而成



数据转换

Old Gray	New Gray	Waveform Content		
…	…	…	…	…
Gray X	Gray Y	…	M	…
…	…	…	…	…

波形表

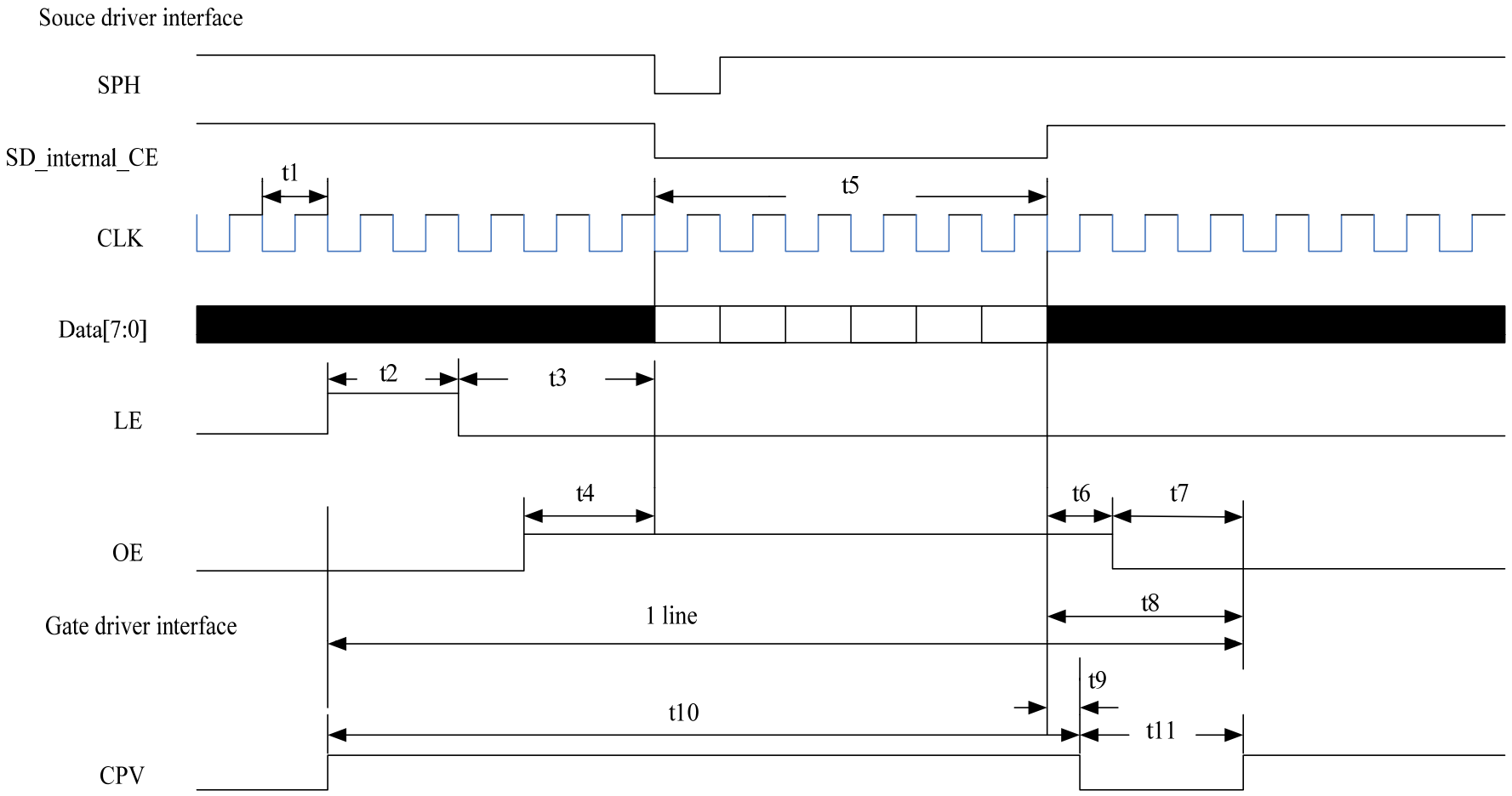


Frame N

对这一个像素而言，上一幅图像里它是灰阶X，现在要刷新到灰阶Y，并且现在刷到第N个frame，则查波形表可知要送的值是M

Frame N

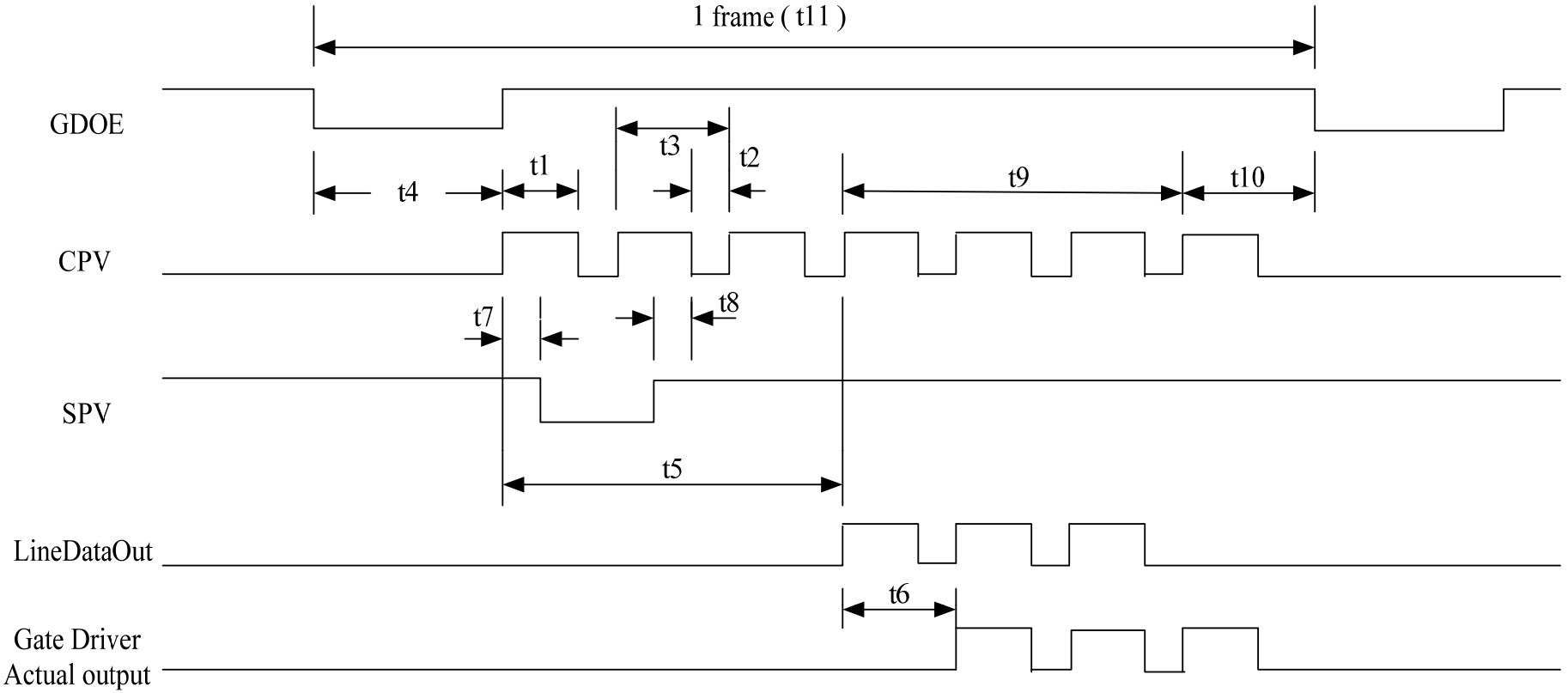
Source时序（一）


Souce driver interface
SPH
SD_internal_CE
t1
t5
CLK
Data[7:0]
t2
t3
LE
t4
t6
t7
OE
Gate driver interface
1 line
t8
t9
t10
t11
CPV

Source时序（二）

Timing	Description	Typical Value	Min	Max
t1	CLK period	125ns	40 ns	-
t2	LE high period	10*t1	t1	-
t3	LE low to SPH low	4*t1	40ns	-
t4	OE high to SPH low	t1	t1	t1
t5	SD_internal_CE active period	200*t1	-	-
t6	SD_internal_CE high to OE low	41*t1	0	-
t7	OE low to CPV high	3*t1	t1	-
t8	SD_internal_CE high to CPV high	t6+t7	-	-
t9	SD_internal_CE high to CPV low	t1	t1	t1
t10	CPV high period	t2+t3+t5+t9	-	-
t11	CPV low period	t8-t9	-	-

Gate时序（一）


1 frame (t11)
GDOE
t4
t1
t3
t2
t9
t10
CPV
t7
t8
SPV
t5
LineDataOut
t6
Gate Driver
Actual output

Gate时序（二）

Timing	Description	Typical Value	Min	Max
t1	CPV high period	see table above	-	-
t2	CPV low period	see table above	-	-
t3	CPV period	t1+t2	-	-
t4	GDOE low period	5*t3	2*t3	-
t5	first CPV to first LineDataOut	3*t3	0	-
t6	Gate driver's actual output valid	t3	-	-
t7	CPV to SPV low	t1/2	t1/2	t1/2
t8	SPV high to CPV	t1/2	t1/2	t1/2
t9	frame data period	600*t3	-	-
t10	frame end period	12*t3	0	-
t11	full frame period	t4+t5+t9+t10	20ms	20ms

驱动EPD屏幕示例伪代码

```
// 按照顺序打开EPD各路高压、VCOM
EPD_Power_Up();
// 需要驱动FRAME_LEN个frame才算更新完毕
//这个FRAME_LEN是由波形表决定的
for(frame=0; frame<FRAME_LEN; frame++)
{
    // frame开始gate的时序
    EPD_Start_Scan();
    // gate扫描，以800×600分辨率为例，gate有
    // 600列，扫完一次600列为一个frame
    for(line=0; line<600; line++)
    {
        // 向source发送一行数据，数据转换也在里面完成
        EPD_Send_Row_Data();
    }
    //frame结束gate的时序
     EPD_End_Scan();
}
// 更新完毕，关闭各路高压、VCOM
EPD_Power_Down();
```

MCU编程时注意事项（一）

- 以800X600分辨率为例，一行有800个像素，一个像素需要2个bit(0为不驱动，1为黑，2为白)，EPD屏接口的数据线为8位，即每个Source CLK送8/2=4个像素，因此送一行数据时需要循环800/4=200次

- 图像数据储存在FLASH里，现EPD用作4灰阶显示，即分为黑，灰阶 1，灰阶2，白这4个状态，因此图像数据里1个像素用2个bit表示（0为黑，1为灰阶1，2为灰阶2，3为白），故800x600的4灰阶图像所占用的FLASH空间为800*600/4=120000字节。

- 由于MCU本身RAM不大，故需要频繁到FLASH里读图像数据并进行数据转换，且MCU速度也不够快，这就导致EPD的刷新一帧的时间比较长，导致最终显示效果异常。为了节省RAM，在图像更新时分为2步，先读上一幅图像数据通过数据转换先把屏驱动到初始状态，再读待更新图像数据通过数据转换把屏驱动到最终的显示效果。为了节省数据转换所占用的时间，在MCU上电后就在RAM生成2个波形表（上一幅图像到初始状态表和初始状态到最终显示图像表）。

  上一幅图像到初始状态表用来表示4个像素图像数据（1个字节）从所有灰阶组合到初始状态经过所有帧的查询 表

MCU编程时注意事项（二）

初始状态到最终显示图像表用来表示4个像素图像数据（1个字节）从初始状态到所有灰阶组合经过所有帧的查询表
因此，在读取图像数据时可以一个字节一个字节直接到RAM里查表而得到最终给EPD数据

- 有了上面两个表，在驱动一帧时可以先从FLASH里读一行(800/4=200字节)上一幅图像数据，通过查上一幅图像到初始状态表进行数据转换后在RAM里缓存转换后的1行（200字节）数据，再调用EPD_Send_Row_Data()函数，通过驱动一定帧数后达到初始状态，再从FLASH里读一行(800/4=200字节)待更新图像数据，通过查初始状态到最终显示图像表进行数据转换后在RAM里缓存转换后的1行（200字节）数据，再调用EPD_Send_Row_Data()函数，通过驱动一定帧数后达到最终的显示效果。

- EPD现在用作4灰阶显示，一帧的时间需要在85ms以内，不同的帧时间需要用不同的波形表，例程中的一帧时间约为82ms，Gate Clock约为7.4KHz，Source Clock为约为3.2MHz。在调试驱动程序时，不要在一帧的时间比较长的情况下连接EDP屏，有可能会造成EDP损坏。

MCU编程时注意事项（三）

- 由于EPD和普通的TFT显示原理不一样，EPD更新图像时需要与上一幅图像作对比，才能达到最终的显示效果。因此，FLASH至少需要保存2张图像数据，即（800*600/4）*2=240000字节。图像更新时需要通过波形表来查询确定EPD某一帧某一像素的驱动方式， 而为了减少一帧的时间，故这个波形 表存放在RAM中，由上面可知，需要2个查询表，如果按例程中一帧时间为82ms时，2个查询表一共有25帧，且该表是与相邻4个像素一起查询，即1个字节（256种可能），故此占用RAM为 25*256=6400字节。
  因此，硬件的资源需要充分满足 上面要求，并留有一定余地，以保证在波形表需调整时导致RAM不足。

  当然在MCU速度够快时也可以不用在RAM中生成波形查询表，而直接在代码中进行数据处理并转换。