(1) Finish the time display function supporting 24-hour (00-23).

1.1 Can display as hour:minute and second, and use a push button or DIP switch to switch the display.

1.2 Support two modes: AM/PM and 24-hour.

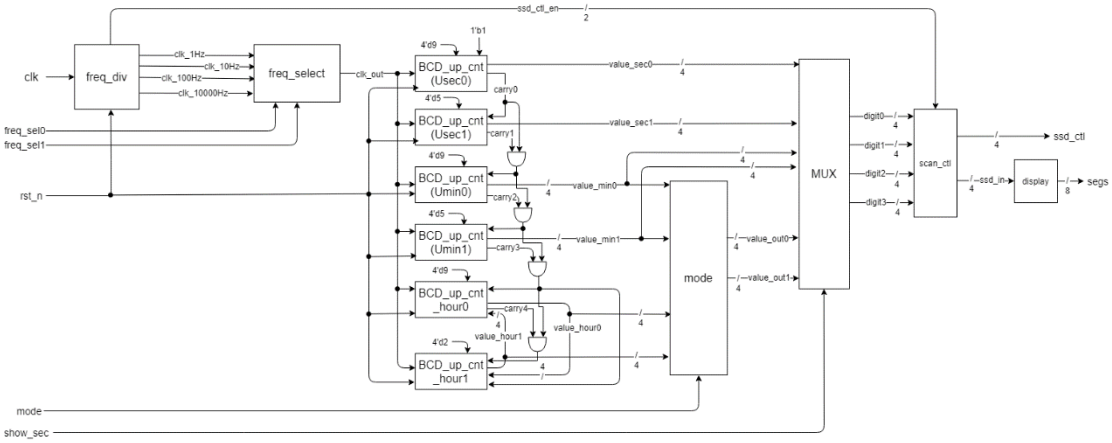
IO:

Input: clk, rst_n, mode, show_sec, freq_sel0, freq_sel1

Output: [7:0]segs, [3:0]ssd_ctl

Block diagram:

本次實驗需要使用以下 module 功能，分別為除頻器(freq_div)、頻率選擇(freq_select)、十進位上數器*6(BCD_up_cnt)、mode 選擇(mode)、多工器(MUX)、scan control(scan_ctl)以及七段顯示解碼器(display)。



1. 除頻器: 由於實驗 demo 時需要將上數器快轉來觀察秒、分、時的連動狀況，因此將 100MHz 的 clk 輸入通過除頻器除頻成 1 Hz(正常)、10Hz(觀察秒)、100Hz(觀察分)以及 10000Hz(觀察小時)的輸出頻率，也輸出 2bit 的 ssd control enable 作為 scan control 的控制項。

2. 頻率選擇: 將除頻器輸出的 4 種頻率透過兩個 DIP switch(freq_sel0、freq_sel1)開關選擇其中一個頻率輸出。

3. 十進位上數器(秒*2、分*2、小時*2): 依據頻率選擇後得到的頻率快慢與上級的進位來進行上數。

4. mode 選擇: 在預設狀況下為 24 小時制，若將開關(mode)打開，則變換成 12 小時制。

5. MUX: 透過一個開關(show_sec)來控制要顯示「小時-分鐘」或是顯示「秒鐘」，預設情形為顯示「小時-分鐘」，若將開關打開，則顯示「秒鐘」。

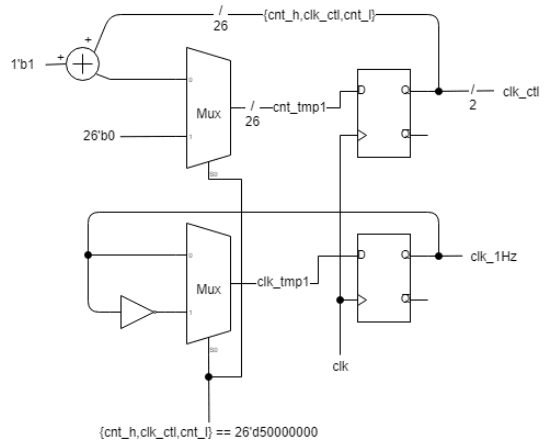
6. scan control: 將 MUX 的輸出 digit3～0 以及除頻器的輸出 ssd control enable 作為輸入，透過快速的輪流顯示，可以讓 4 個七段顯示器各自顯示自己要顯示的數字，並讓眼睛看起來是 4 個顯示器同時顯示的。

7.七段顯示解碼器: 將 scan control 給的輸入值(ssd_in)透過解碼之後顯示在七段顯示器上。

Logic diagram:

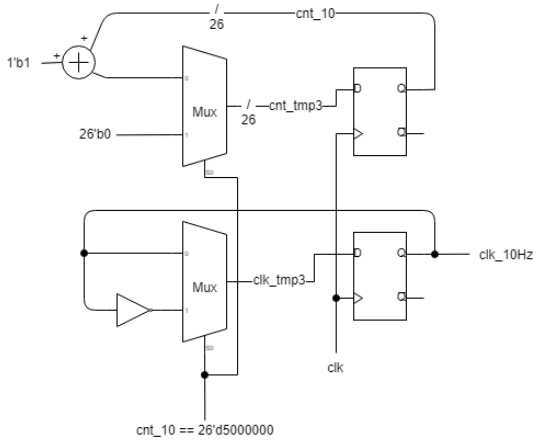
<u>除頻器:</u> 輸出 1Hz、10Hz、100Hz、10000Hz 以及 2bit 的 clk_ctl 控制

<u>1Hz:</u>



由於 FPGA 板上的頻率為 100MHz，建構理念是設計一個上限為 50M 的 binary up counter，在達到 50M 之前，每個 clk 都讓 counter 加上 1，而當 counter 數到 50M 時，便將輸出值(clk_1Hz) toggle 一次，也將 counter 歸零，如此下來，輸出值(clk_1Hz)在等於 0 和等於 1 的時間各自都是 50M 次的 clk，也就是說，clk_1Hz 的值 0.5 秒會 toggle 一次，因此輸出值的頻率將會等於 100MHz/(50M*2) = 1Hz。
另外，將 counter 中的第 16、17bit 抓出來當作給 scan control 的控制項。

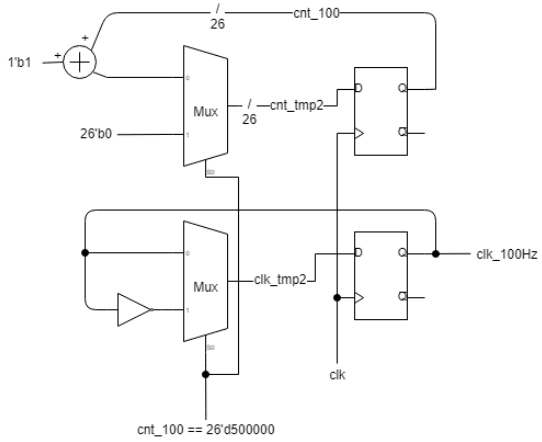
<u>10Hz:</u> 觀察秒



由於 FPGA 板上的頻率為 100MHz，建構理念是設計一個上限為 5M 的 binary up counter，在達到 5M 之前，每個 clk 都讓 counter 加上 1，而當 counter 數到 5M

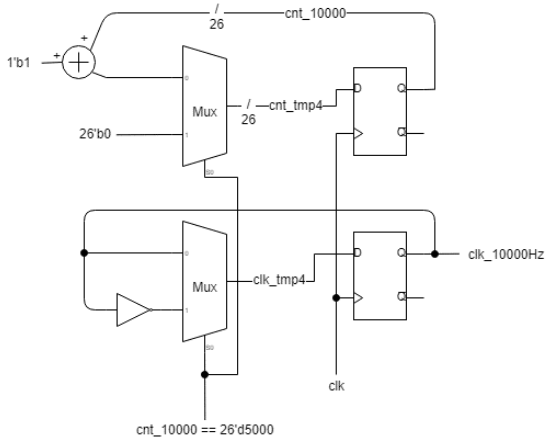
時，便將輸出值(clk_10Hz) toggle 一次，也將 counter 歸零，如此下來，輸出值(clk_10Hz)在等於 0 和等於 1 的時間各自都是 5M 次的 clk，也就是說，clk_10Hz 的值 0.05 秒會 toggle 一次，因此輸出值的頻率將會等於 100MHz/(5M*2) = 10Hz。

100Hz: 觀察分



由於 FPGA 板上的頻率為 100MHz，建構理念是設計一個上限為 500k 的 binary up counter，在達到 500k 之前，每個 clk 都讓 counter 加上 1，而當 counter 數到 500k 時，便將輸出值(clk_100Hz) toggle 一次，也將 counter 歸零，如此下來，輸出值(clk_100Hz)在等於 0 和等於 1 的時間各自都是 500k 次的 clk，也就是說，clk_100Hz 的值 0.005 秒會 toggle 一次，因此輸出值的頻率將會等於 100MHz/(500k*2) = 100Hz。

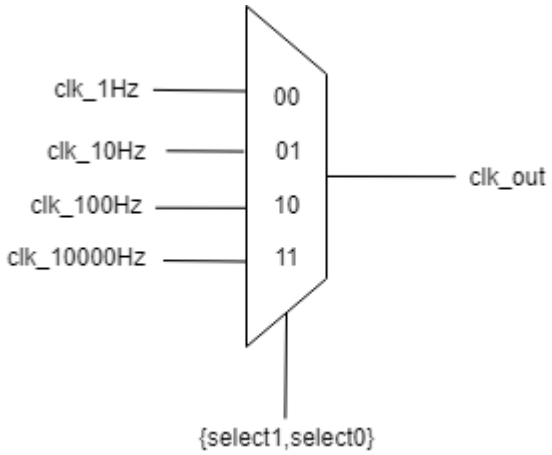
10000Hz: 觀察小時



由於 FPGA 板上的頻率為 100MHz，建構理念是設計一個上限為 5k 的 binary up counter，在達到 5k 之前，每個 clk 都讓 counter 加上 1，而當 counter 數到 5k 時，便將輸出值(clk_10000Hz) toggle 一次，也將 counter 歸零，如此下來，輸出值(clk_10000Hz)在等於 0 和等於 1 的時間各自都是 5k 次的 clk，也就是說，

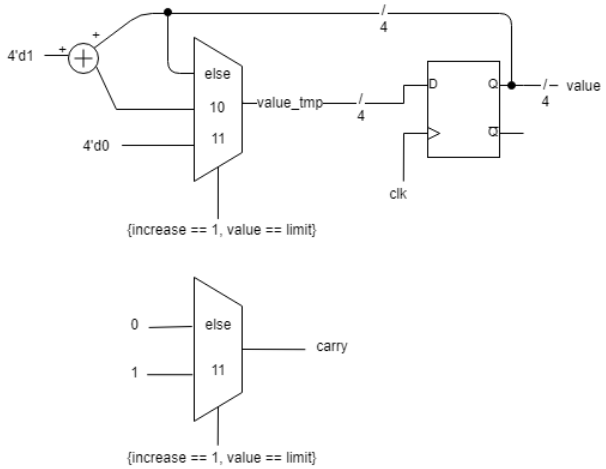
clk_10000Hz 的值 0.00005 秒會 toggle 一次，因此輸出值的頻率將會等於 100MHz/(5k*2) = 10000Hz。

<u>頻率選擇:</u> 將除頻器輸出的 4 種頻率透過兩個 DIP switch(freq_sel0、freq_sel1) 開關選擇其中一個頻率輸出。



當兩個開關為 00(預設)時 → 輸出 1Hz 頻率(clk_out = clk_1Hz) (正常)；
當兩個開關為 01 時 → 輸出 10Hz 頻率(clk_out = clk_10Hz) (觀察秒)；
當兩個開關為 10 時 → 輸出 100Hz 頻率(clk_out = clk_100Hz) (觀察分)；
當兩個開關為 11 時 → 輸出 10000Hz 頻率(clk_out = clk_10000Hz) (觀察時)。

<u>十進位上數器(秒*2、分*2、小時*2):</u> 依據頻率選擇後得到的頻率快慢與上級的進位來進行上數。

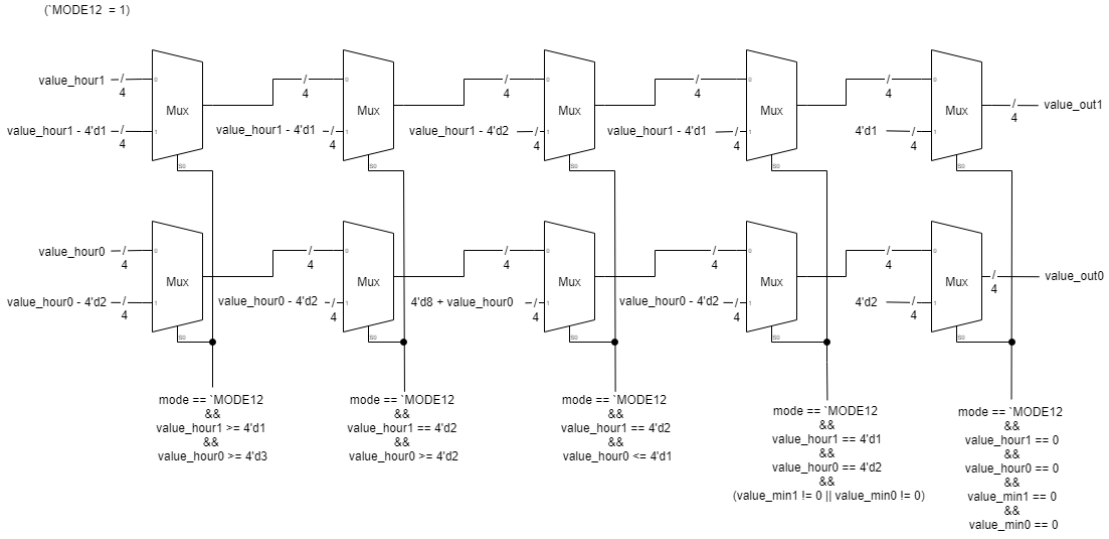


秒數個位數: (increase 為 1, limit 為 9)
秒數十位數: (increase 為秒數個位數的 carry, limit 為 5)
分鐘個位數: (increase 為秒數個位和十位數的 carry(皆為 1 時), limit 為 9)
分鐘十位數: (increase 為秒數個位、十位數以及分鐘個位數的 carry(皆為 1 時),

limit 為 5)
小時個位數: (increase 為秒數個位、十位數以及分鐘個位、十位數的 carry(皆為 1 時), limit 為 9)
小時十位數: (increase 為秒數個位、十位數和分鐘個位、十位數以及小時個位數的 carry(皆為 1 時), limit 為 2)

當 increase 等於 1 且 value 等於 limit 時，代表下次需要進位，carry 輸出 1 告知下一級，value_tmp = 0；當 increase 等於 1 但 value 不等於 limit 時，代表下次要加 1 但不用進位，carry = 0，value_tmp = value + 1；其他狀況代表不需要加 1 也不需要進位，carry = 0，value 維持現狀，value_tmp = value。當每次 clk 的 positive edge 來時，將 value 輸入 value_tmp 的值。
在小時的上數器中，由於小時上限為 23(0～23)，因此另外再加上一個判斷為當小時十位數 = 2 且小時個位數 = 3 時，小時的 value_tmp 都 = 0，因此下一個 clk 時小時的 value 都會輸入 0，重新從 0 開始數。

<u>mode 選擇:</u> 在預設狀況下為 24 小時制，若將開關(mode)打開，則變換成 12 小時制。

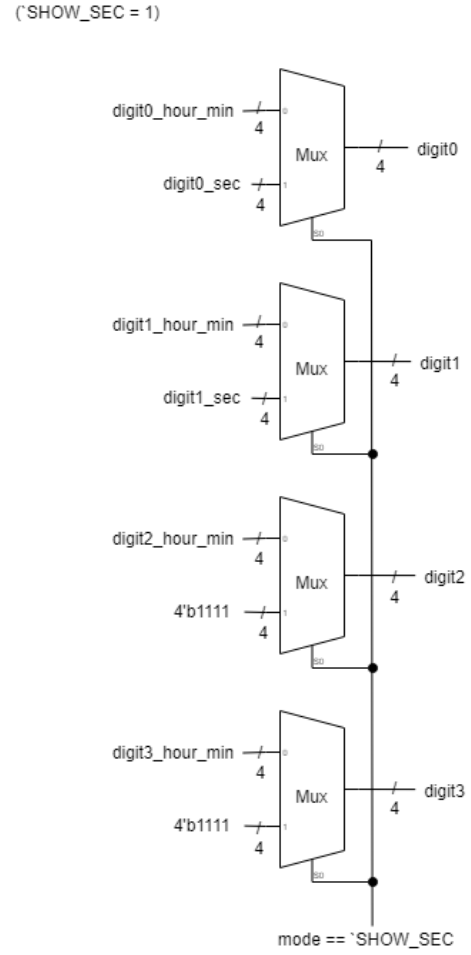


value_hour1: 24 小時制小時十位數輸入值
value_hour0: 24 小時制小時個位數輸入值
value_out1: 小時十位數輸出值
value_out0: 小時個位數輸出值
(1)當 mode = 1、小時為 00、分鐘為 00 → 現在為午夜 00:00(AM12:00)且需要轉換成 12 小時制 → 十位數 = 1, 個位數 = 2。
(2)當 mode = 1、小時為 12、分鐘不為 00 且不符合(1)時 → 現在為中午 12:01～12:59(PM00:01～00:59)且需要轉換成 12 小時制 → 十位數減 1, 個位數減 2。

(3)當 mode = 1、小時為 20～21 且不符合(1)跟(2)時 → 現在為晚上 20:00～21:59(PM08:00～09:59)且需要轉換成 12 小時制但個位數不夠減 → 十位數減 2, 個位數加 8。
(4)當 mode = 1、小時為 22～23 且不符合(1)～(3)時 → 現在為晚上 22:00～23:59(PM10:00～11:59)且需要轉換成 12 小時制且個位數夠減 → 十位數減 1, 個位數減 2。
(5)當 mode = 1、小時為 13～19 且不符合(1)～(4)時 → 現在為 13:00～19:59(PM01:00～07:59)且需要轉換成 12 小時制 → 十位數減 1, 個位數減 2。

經過以上(1)～(5)之後，所有需要轉換的部分都已經處理完畢，因此
(6)其餘狀況: 輸出值等於輸入值。

MUX: 透過一個開關(show_sec)來控制要顯示「小時-分鐘」或是顯示「秒鐘」，預設情形為顯示「小時-分鐘」，若將開關打開，則顯示「秒鐘」。



以 mode 連接 DIP switch 開關(show_sec)訊號，digit3～2_hour_min 連接 mode 的輸出值(小時)，digit1～0_hour_min 連接分鐘上數器的輸出值(分鐘)，digit1～0_sec 連接秒數上數器的輸出值(秒數)，當 mode = 1 時，要顯示「秒鐘」；mode = 0 時，要顯示「小時-分鐘」，透過 mode 訊號選擇輸出訊號 digit3～0 的值，並

輸出至 scan control 當作輸入。由於秒鐘只有兩位數，因此在顯示「秒鐘」時，4’b1111 代表的是讓兩個七段顯示器不顯示。

Scan control: 將 MUX 的輸出 digit3～0 以及除頻器的輸出 ssd control enable 作為輸入，透過快速的輪流顯示，可以讓 4 個七段顯示器各自顯示自己要顯示的數字，並讓眼睛看起來是 4 個顯示器同時顯示的。

ssd_ctl_en = 00 → ssd_ctl = 0111
ssd_ctl_en = 01 → ssd_ctl = 1011
ssd_ctl_en = 10 → ssd_ctl = 1101
ssd_ctl_en = 11 → ssd_ctl = 1110

四個七段顯示器輪流顯示，ssd_ctl_en = 00 時→顯示第一個七段顯示器(依據模式顯示 1.小時十位數 2.不亮)；ssd_ctl_en = 01 時→顯示第二個七段顯示器(依據模式顯示 1.小時個位數 2.不亮)；ssd_ctl_en = 10 時→顯示第三個七段顯示器(依據模式顯示 1.分鐘十位數 2.秒數十位數)；ssd_ctl_en = 11 時→顯示第四個七段顯示器(依據模式顯示 1.分鐘個位數 2.秒數個位數)，以快速輪流顯示的方式達成視覺暫留的效果。

7-segment display decoder: 將 scan control 給的輸入值(ssd_in)透過解碼之後顯示在七段顯示器上。

in = 0 → segs = 8'b0000_0011
in = 1 → segs = 8'b1001_1111
in = 2 → segs = 8'b0010_0101
in = 3 → segs = 8'b0000_1101
in = 4 → segs = 8'b1001_1001
in = 5 → segs = 8'b0100_1001
in = 6 → segs = 8'b0100_0001
in = 7 → segs = 8'b0001_1111
in = 8 → segs = 8'b0000_0001
in = 9 → segs = 8'b0000_1001
default 為 segs = 8’b1111_1111(全暗)

結合以上功能，可以得到一個支援 24 小時制/12 小時制，且可以切換顯示「小時-分鐘」或是顯示「秒鐘」功能的電子鐘。

I/O pin assignment:

segs[7]	segs[6]	segs[5]	segs[4]	segs[3]	segs[2]	segs[1]	segs[0]
W7	W6	U8	V8	U5	V5	U7	V7

ssd_ctl[3]	ssd_ctl[2]	ssd_ctl[1]	ssd_ctl[0]	clk	rst_n
W4	V4	U4	U2	W5	R2

mode	show_sec	freq_sel0	freq_sel1
V16	V17	U1	T1

(2) For the date functions in clock (no leap year), we have the following functions:

o Day (Jan/March/May/July/Aug/Oct/Dec: 1-31, Feb: 28, Apr/June/Sept/Nov: 30),

o Month (1-12),

o Year (00-99).

Implement the following functions:

2.1 Month-Day function display in the 4 7-segment displays.

2.2 Combine the Year and 1.1 to finish a Year-Month-Day, and use one DIP switch to select the display of Year (2 Seven-Segment Displays, SSDs) or Month-Day (4 SSDs).

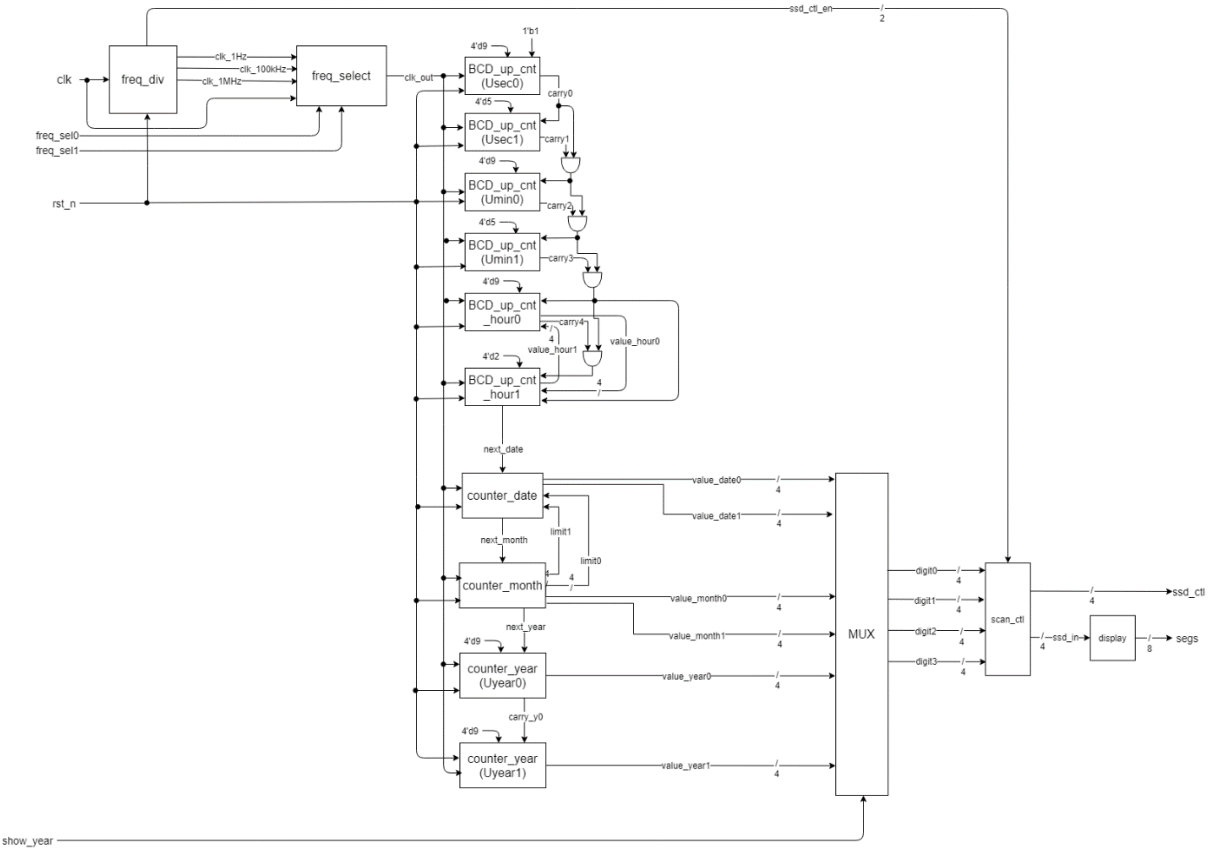
IO:

Input: clk, rst_n, show_year, freq_sel0, freq_sel1

Output: [7:0]segs, [3:0]ssd_ctl

Block diagram:

本次實驗需要使用以下 module 功能，分別為除頻器(freq_div)、頻率選擇(freq_select)、十進位上數器*6(BCD_up_cnt)、日期上數器(counter_date)、月份上數器(counter_month)、年份上數器*2 (counter_year)、多工器(MUX)、scan control(scan_ctl)以及七段顯示解碼器(display)。



1. 除頻器: 由於實驗 demo 時需要將上數器快轉來觀察日、月、年的連動狀

況，因此將 100MHz 的 clk 輸入通過除頻器除頻成 1 Hz(正常)、100kHz(觀察日)、1MHz(觀察月)的輸出頻率，也輸出 2bit 的 ssd control enable 作為 scan control 的控制項。

2. 頻率選擇: 將除頻器輸出的 3 種頻率加上未經除頻的 100MHz 頻率(觀察年)透過兩個 DIP switch(freq_sel0、freq_sel1)開關選擇其中一個頻率輸出。

3. 十進位上數器(秒*2、分*2、小時*2): 依據頻率選擇後得到的頻率快慢與上級的進位來進行上數。

4. 日期上數器: 依據頻率選擇後得到的頻率快慢與上級的進位來進行上數(數到月份上數器所給的的日數上限)。

5. 月份上數器: 依據頻率選擇後得到的頻率快慢與上級的進位來進行上數；另外，判斷當前的月份得到當月的總日數，並告知日期上數器。

6. 年份上數器*2: 依據頻率選擇後得到的頻率快慢與上級的進位來進行上數。

7. MUX: 透過一個開關(show_year)來控制要顯示「月-日」或是顯示「年」，預設情形為顯示「月-日」，若將開關打開，則顯示「年」。

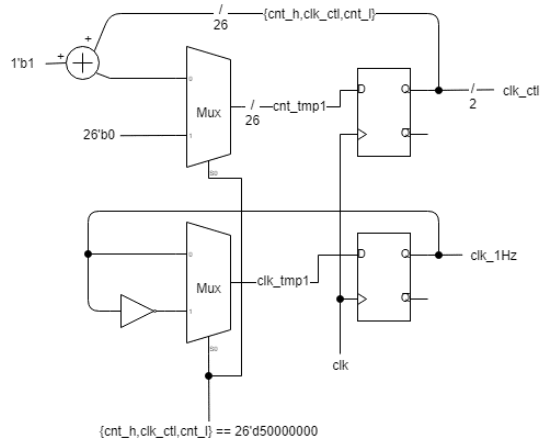
8. scan control: 將 MUX 的輸出 digit3～0 以及除頻器的輸出 ssd control enable 作為輸入，透過快速的輪流顯示，可以讓 4 個七段顯示器各自顯示自己要顯示的數字，並讓眼睛看起來是 4 個顯示器同時顯示的。

9.七段顯示解碼器: 將 scan control 給的輸入值(ssd_in)透過解碼之後顯示在七段顯示器上。

Logic diagram:

<u>除頻器:</u> 輸出 1Hz、100kHz、1MHz 以及 2bit 的 clk_ctl 控制

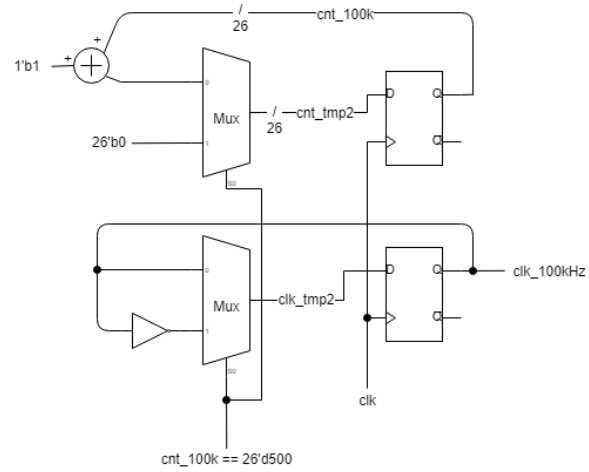
<u>1Hz:</u>



由於 FPGA 板上的頻率為 100MHz，建構理念是設計一個上限為 50M 的 binary up counter，在達到 50M 之前，每個 clk 都讓 counter 加上 1，而當 counter 數到 50M 時，便將輸出值(clk_1Hz) toggle 一次，也將 counter 歸零，如此下來，輸出值(clk_1Hz)在等於 0 和等於 1 的時間各自都是 50M 次的 clk，也就是說，clk_1Hz

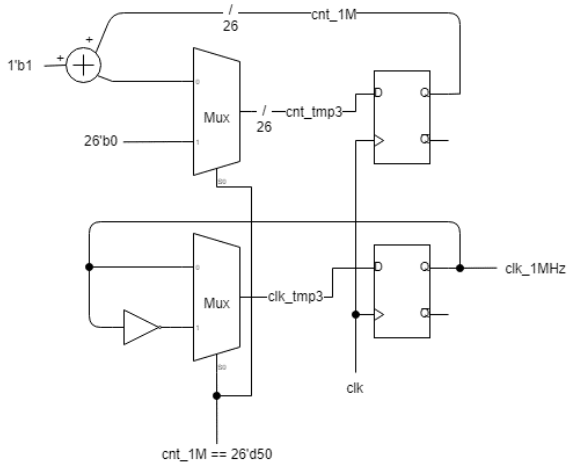
的值 0.5 秒會 toggle 一次，因此輸出值的頻率將會等於 100MHz/(50M*2) = 1Hz。
另外，將 counter 中的第 16、17bit 抓出來當作給 scan control 的控制項。

100kHz: 觀察日



由於 FPGA 板上的頻率為 100MHz，建構理念是設計一個上限為 500 的 binary up counter，在達到 500 之前，每個 clk 都讓 counter 加上 1，而當 counter 數到 500 時，便將輸出值(clk_100kHz) toggle 一次，也將 counter 歸零，如此下來，輸出值(clk_100kHz)在等於 0 和等於 1 的時間各自都是 500 次的 clk，也就是說，clk_100kHz 的值 0.000005 秒會 toggle 一次，因此輸出值的頻率將會等於 100MHz/(500*2) = 100kHz。

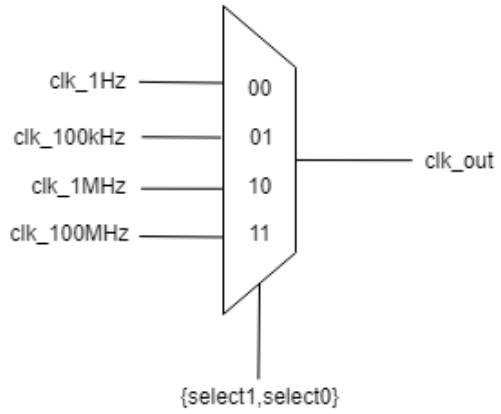
1MHz: 觀察月



由於 FPGA 板上的頻率為 100MHz，建構理念是設計一個上限為 50 的 binary up counter，在達到 50 之前，每個 clk 都讓 counter 加上 1，而當 counter 數到 50 時，便將輸出值(clk_1MHz) toggle 一次，也將 counter 歸零，如此下來，輸出值

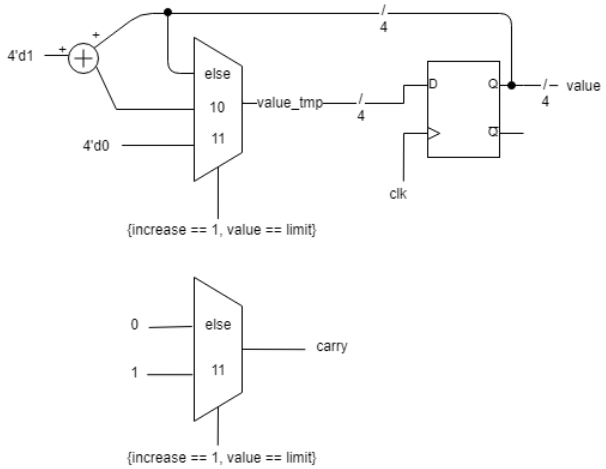
(clk_1MHz)在等於 0 和等於 1 的時間各自都是 50 次的 clk，也就是說，clk_1MHz 的值 0.0000005 秒會 toggle 一次，因此輸出值的頻率將會等於 100MHz/(50*2) = 1MHz。

<u>頻率選擇:</u> 將除頻器輸出的 3 種頻率加上未經除頻的 100MHz 頻率(觀察年)透過兩個 DIP switch(freq_sel0、freq_sel1)開關選擇其中一個頻率輸出。



當兩個開關為 00(預設)時 → 輸出 1Hz 頻率(clk_out = clk_1Hz) (正常)；
當兩個開關為 01 時 → 輸出 100kHz 頻率(clk_out = clk_100kHz) (觀察日)；
當兩個開關為 10 時 → 輸出 1MHz 頻率(clk_out = clk_1MHz) (觀察月)；
當兩個開關為 11 時 → 輸出 100MHz 頻率(clk_out = clk_100MHz)(觀察年)。

<u>十進位上數器(秒*2、分*2、小時*2):</u> 依據頻率選擇後得到的頻率快慢與上級的進位來進行上數。

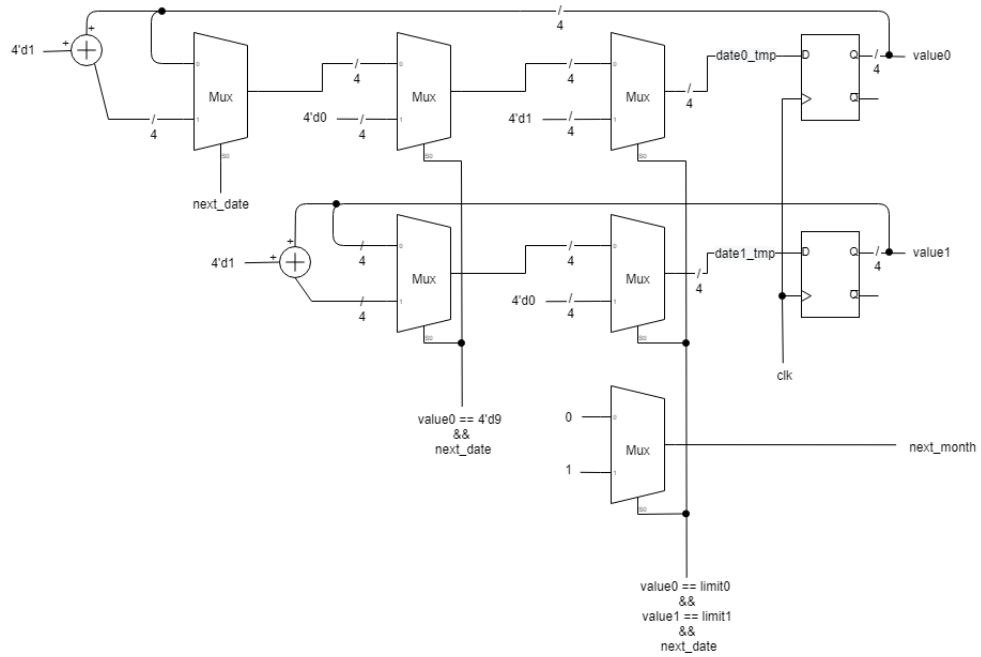


秒數個位數: (increase 為 1, limit 為 9)
秒數十位數: (increase 為秒數個位數的 carry, limit 為 5)
分鐘個位數: (increase 為秒數個位和十位數的 carry(皆為 1 時), limit 為 9)
分鐘十位數: (increase 為秒數個位、十位數以及分鐘個位數的 carry(皆為 1 時),

limit 為 5)
小時個位數: (increase 為秒數個位、十位數以及分鐘個位、十位數的 carry(皆為 1 時), limit 為 9)
小時十位數: (increase 為秒數個位、十位數和分鐘個位、十位數以及小時個位數的 carry(皆為 1 時), limit 為 2)

當 increase 等於 1 且 value 等於 limit 時，代表下次需要進位，carry 輸出 1 告知下一級，value_tmp = 0；當 increase 等於 1 但 value 不等於 limit 時，代表下次要加 1 但不用進位，carry = 0，value_tmp = value + 1；其他狀況代表不需要加 1 也不需要進位，carry = 0，value 維持現狀，value_tmp = value。當每次 clk 的 positive edge 來時，將 value 輸入 value_tmp 的值。
在小時的上數器中，由於小時上限為 23(0～23)，因此另外再加上一個判斷為當小時十位數 = 2 且小時個位數 = 3 時，小時的 value_tmp 都 = 0，因此下一個 clk 時小時的 value 都會輸入 0，重新從 0 開始數。

<u>日期上數器:</u> 依據頻率選擇後得到的頻率快慢與上級的進位來進行上數(數到月份上數器所給的的日數上限)。



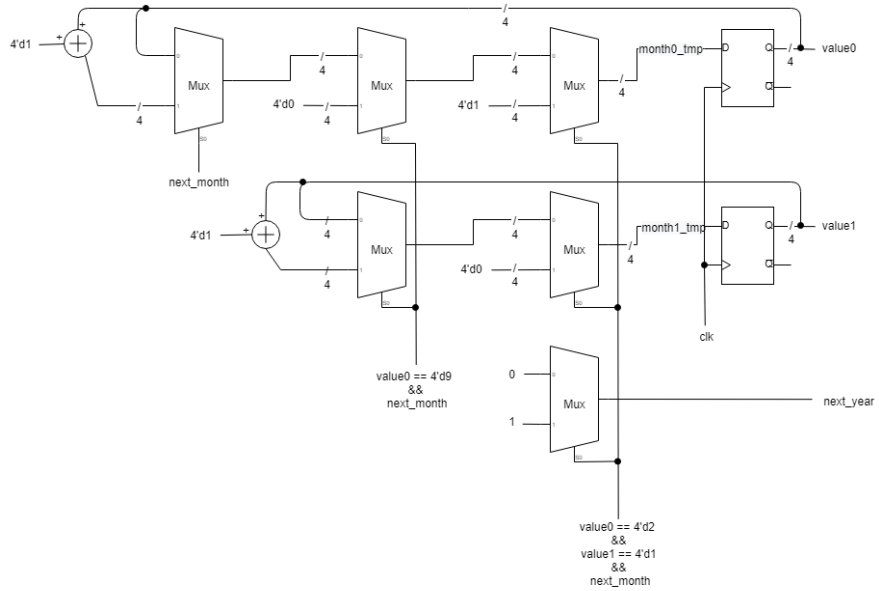
next_date 為(時、分、秒)所有上數器的 carry 都等於 1 時，也就是日期要加 1 的時候的通知信號。
value0 是日期個位數、value1 是日期十位數。
limit1、limit0 為當月日數上限的十位數與個位數。

(1)當 value0 = limit0、value1 = limit1、next_date = 1 時 → 下一天為下個月的 01

日 → date1_tmp = 0 , date0_tmp = 1。
(2)當 value0 = 9、next_date = 1 且不符合(1)時 → 個位數進位、十位數加 1 → date1_tmp = value1 + 1 , date0_tmp = 0。
(3)當 next_date = 1 且不符合(1)跟(2)時 → 個位數加 1、十位數不變 → date1_tmp = value1 , date0_tmp = value0 + 1。
(4)其餘狀況為上級未進位 → 日期不需要變動。
當每次 clk 的 positive edge 來時，將各 value 輸入各 value_tmp 的值。

當 value0 = limit0、value1 = limit1、next_date = 1 時 → 下一天為下個月的 01 日 → 告知月份上數器要加 1 → next_month = 1；其餘狀況月份不需要加 1 → next_month = 0。

<u>月份上數器:</u> 依據頻率選擇後得到的頻率快慢與上級的進位來進行上數；另外，判斷當前的月份得到當月的總日數，並告知日期上數器。



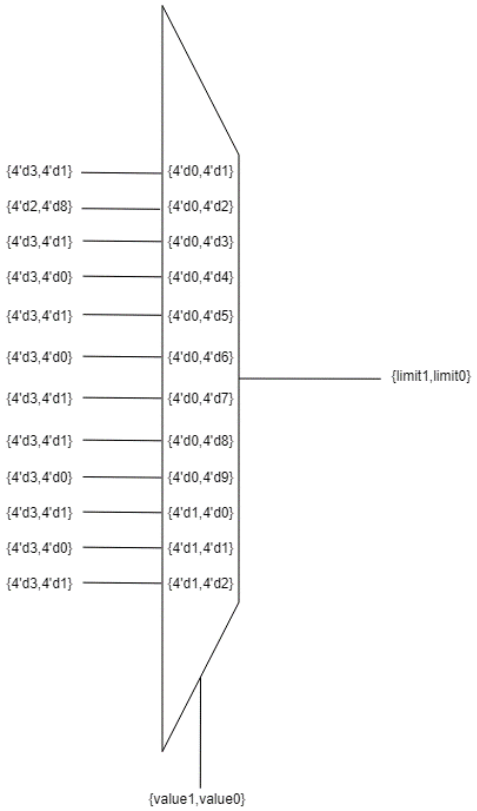
next_month 為(時、分、秒、日)所有上數器的 carry 都等於 1 時，也就是月份要加 1 的時候的通知信號。
value0 是月份個位數、value1 是月份十位數。

(1)當 value0 = 2、value1 = 1、next_month = 1 時 → 下個月為隔年 01 月→ date1_tmp = 0 , date0_tmp = 1。
(2)當 value0 = 9、next_month = 1 且不符合(1)時 → 個位數進位、十位數加 1 → date1_tmp = value1 + 1 , date0_tmp = 0。
(3)當 next_month = 1 且不符合(1)跟(2)時 → 個位數加 1、十位數不變 → date1_tmp = value1 , date0_tmp = value0 + 1。

(4)其餘狀況為上級未進位 → 月份不需要變動。
當每次 clk 的 positive edge 來時，將各 value 輸入各 value_tmp 的值。

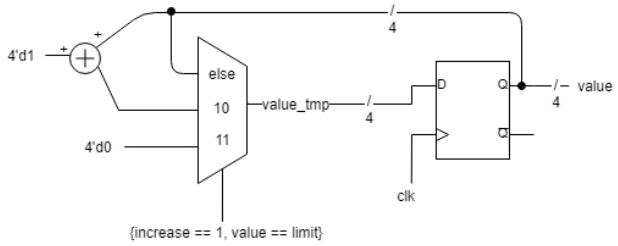
當 value0 = 2、value1 = 1、next_month = 1 時 → 下個月為隔年 01 月 → 告知年份上數器要加 1 → next_year = 1；其餘狀況年份不需要加 1 → next_year = 0。

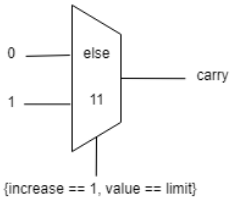
limit1、limit0 為當月日數上限的十位數與個位數。



(1) value1 = 0、value0 = 1 → 1 月 → 共 31 天 → limit1 = 3, limit0 = 1
(2) value1 = 0、value0 = 2 → 2 月 → 共 28 天 → limit1 = 2, limit0 = 8
(3) value1 = 0、value0 = 3 → 3 月 → 共 31 天 → limit1 = 3, limit0 = 1
(4) value1 = 0、value0 = 4 → 4 月 → 共 30 天 → limit1 = 3, limit0 = 0
(5) value1 = 0、value0 = 5 → 5 月 → 共 31 天 → limit1 = 3, limit0 = 1
(6) value1 = 0、value0 = 6 → 6 月 → 共 30 天 → limit1 = 3, limit0 = 0
(7) value1 = 0、value0 = 7 → 7 月 → 共 31 天 → limit1 = 3, limit0 = 1
(8) value1 = 0、value0 = 8 → 8 月 → 共 31 天 → limit1 = 3, limit0 = 1
(9) value1 = 0、value0 = 9 → 9 月 → 共 30 天 → limit1 = 3, limit0 = 0
(10) value1 = 1、value0 = 0 → 10 月 → 共 31 天 → limit1 = 3, limit0 = 1
(11) value1 = 1、value0 = 1 → 11 月 → 共 30 天 → limit1 = 3, limit0 = 0
(12) value1 = 1、value0 = 2 → 12 月 → 共 31 天 → limit1 = 3, limit0 = 1

<u>年份上數器*2:</u> 依據頻率選擇後得到的頻率快慢與上級的進位來進行上數。

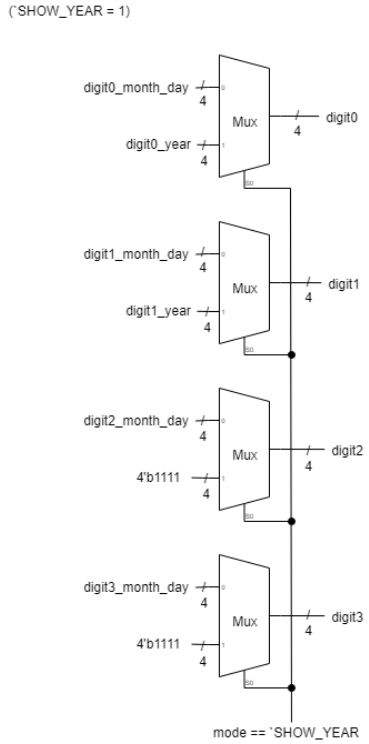




年份個位數: (increase 為月份要進位的信號 next_year, limit 為 9)
年份十位數: (increase 年份個位數的 carry, limit 為 9)

當 increase 等於 1 且 value 等於 limit 時，代表下次需要進位，carry 輸出 1 告知下一級，value_tmp = 0；當 increase 等於 1 但 value 不等於 limit 時，代表下次要加 1 但不用進位，carry = 0，value_tmp = value + 1；其他狀況代表不需要加 1 也不需要進位，carry = 0，value 維持現狀，value_tmp = value。當每次 clk 的 positive edge 來時，將 value 輸入 value_tmp 的值。

<u>MUX:</u> 透過一個開關(show_year)來控制要顯示「月-日」或是顯示「年」，預設情形為顯示「月-日」，若將開關打開，則顯示「年」。



以 mode 連接 DIP switch 開關(show_year)訊號，digit3～2_ month_day 連接月份上數器的輸出值，digit1～0_ month_day 連接日期上數器的輸出值，digit1～0_year 連接年份上數器的輸出值，當 mode = 1 時，要顯示「年」；mode = 0 時，要顯示「月-日」，透過 mode 訊號選擇輸出訊號 digit3～0 的值，並輸出至 scan control 當作輸入。由於年只要顯示後兩位數，因此在顯示「年」時，4'b1111 代表的是讓兩個七段顯示器不顯示。

Scan control: 將 MUX 的輸出 digit3～0 以及除頻器的輸出 ssd control enable 作為輸入，透過快速的輪流顯示，可以讓 4 個七段顯示器各自顯示自己要顯示的數字，並讓眼睛看起來是 4 個顯示器同時顯示的。

ssd_ctl_en = 00 → ssd_ctl = 0111
ssd_ctl_en = 01 → ssd_ctl = 1011
ssd_ctl_en = 10 → ssd_ctl = 1101
ssd_ctl_en = 11 → ssd_ctl = 1110

四個七段顯示器輪流顯示，ssd_ctl_en = 00 時→顯示第一個七段顯示器(依據模式顯示 1.月份十位數 2.不亮)；ssd_ctl_en = 01 時→顯示第二個七段顯示器(依據模式顯示 1.月份個位數 2.不亮)；ssd_ctl_en = 10 時→顯示第三個七段顯示器(依據模式顯示 1.日期十位數 2.年份十位數)；ssd_ctl_en = 11 時→顯示第四個七段顯示器(依據模式顯示 1.日期個位數 2. 年份個位數)，以快速輪流顯示的方式達成視覺暫留的效果。

7-segment display decoder: 將 scan control 給的輸入值(ssd_in)透過解碼之後顯示在七段顯示器上。

in = 0 → segs = 8'b0000_0011
in = 1 → segs = 8'b1001_1111
in = 2 → segs = 8'b0010_0101
in = 3 → segs = 8'b0000_1101
in = 4 → segs = 8'b1001_1001
in = 5 → segs = 8'b0100_1001
in = 6 → segs = 8'b0100_0001
in = 7 → segs = 8'b0001_1111
in = 8 → segs = 8'b0000_0001
in = 9 → segs = 8'b0000_1001
default 為 segs = 8'b1111_1111(全暗)

結合以上功能，可以完成一個可以切換顯示「月-日」或是顯示「年」的電子日歷。

I/O pin assignment:

segs[7]	segs[6]	segs[5]	segs[4]	segs[3]	segs[2]	segs[1]	segs[0]
W7	W6	U8	V8	U5	V5	U7	V7

ssd_ctl[3]	ssd_ctl[2]	ssd_ctl[1]	ssd_ctl[0]	clk	rst_n
W4	V4	U4	U2	W5	R2

show_year	freq_sel0	freq_sel1
V17	U1	T1

(3) (Bonus) Add the time display support of both AM/PM and 24-hour, and the leap year support. (The year will start from 2000 to 2200 and use 4 SSDs to display.)

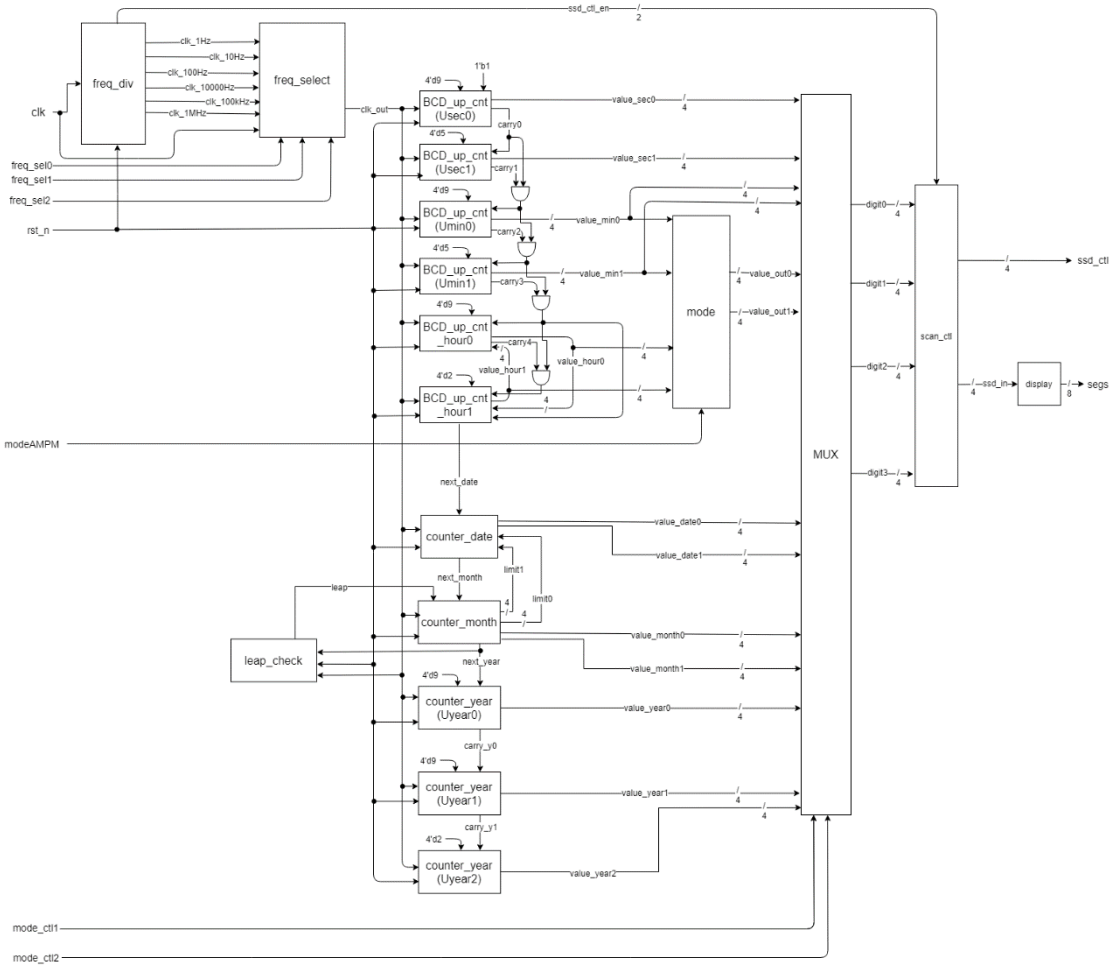
IO:

Input: clk, rst_n, mode_ctl1, mode_ctl2, modeAMPM, freq_sel0, freq_sel1, freq_sel2

Output: [7:0]segs, [3:0]ssd_ctl

Block diagram:

本次實驗需要使用以下 module 功能，分別為除頻器(freq_div)、頻率選擇(freq_select)、十進位上數器*6(BCD_up_cnt)、日期上數器(counter_date)、月份上數器(counter_month)、年份上數器*3 (counter_year)、閏年檢查(leap_check)、mode 選擇(mode)、多工器(MUX)、scan control(scan_ctl)以及七段顯示解碼器(display)。



1. 除頻器: 由於實驗 demo 時需要將上數器快轉來觀察秒、分、時、日、月、年的連動狀況，因此將 100MHz 的 clk 輸入通過除頻器除頻成 1 Hz(正常)、

10Hz(觀察秒)、100Hz(觀察分)、10000Hz(觀察小時)、100kHz(觀察日)以及1MHz(觀察月)的輸出頻率，也輸出 2bit 的 ssd control enable 作為 scan control 的控制項。

2. 頻率選擇: 將除頻器輸出的 6 種頻率加上未經除頻的 100MHz 頻率(觀察年)透過三個 DIP switch(freq_sel0、freq_sel1、freq_sel2)開關選擇其中一個頻率輸出。

3. 十進位上數器(秒*2、分*2、小時*2): 依據頻率選擇後得到的頻率快慢與上級的進位來進行上數。

4. 日期上數器: 依據頻率選擇後得到的頻率快慢與上級的進位來進行上數(數到月份上數器所給的的日數上限)。

5. 月份上數器: 依據頻率選擇後得到的頻率快慢與上級的進位來進行上數；另外，判斷當前的月份得到當月的總日數，並告知日期上數器。

6. 年份上數器*3: 依據頻率選擇後得到的頻率快慢與上級的進位來進行上數。

7. 閏年檢查: 連接通知年份進位的信號當輸入，並計算該年是否為閏年，並將結果告知月份上數器中判斷該月總日數的地方。

8. mode 選擇: 在預設狀況下為 24 小時制，若將開關(modeAMPM)打開，則變換成 12 小時制。

9. MUX: 透過兩個開關(mode_ctl1、mode_ctl2)來控制要顯示「小時-分鐘」或是顯示「秒鐘」或是顯示「月-日」或是顯示「年」，預設情形為顯示「小時-分鐘」。

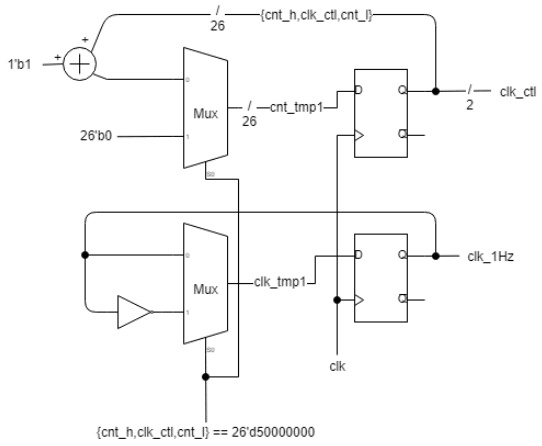
10. scan control: 將 MUX 的輸出 digit3～0 以及除頻器的輸出 ssd control enable 作為輸入，透過快速的輪流顯示，可以讓 4 個七段顯示器各自顯示自己要顯示的數字，並讓眼睛看起來是 4 個顯示器同時顯示的。

11.七段顯示解碼器: 將 scan control 給的輸入值(ssd_in)透過解碼之後顯示在七段顯示器上。

Logic diagram:

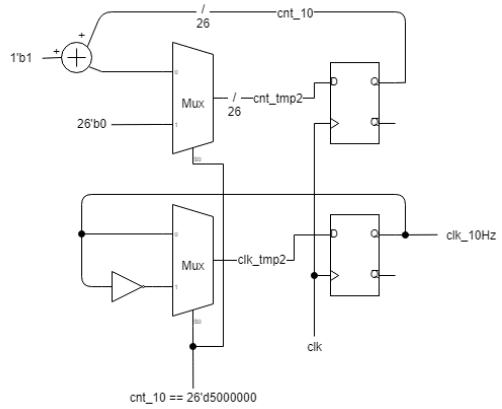
<u>除頻器:</u> 將 100MHz 的 clk 輸入通過除頻器除頻成 1 Hz(正常)、10Hz(觀察秒)、100Hz(觀察分)、10000Hz(觀察小時)、100kHz(觀察日)以及 1MHz(觀察月)的輸出頻率，也輸出 2bit 的 ssd control enable 作為 scan control 的控制項。

<u>1Hz:</u>



由於 FPGA 板上的頻率為 100MHz，建構理念是設計一個上限為 50M 的 binary up counter，在達到 50M 之前，每個 clk 都讓 counter 加上 1，而當 counter 數到 50M 時，便將輸出值(clk_1Hz) toggle 一次，也將 counter 歸零，如此下來，輸出值(clk_1Hz)在等於 0 和等於 1 的時間各自都是 50M 次的 clk，也就是說，clk_1Hz 的值 0.5 秒會 toggle 一次，因此輸出值的頻率將會等於 100MHz/(50M*2) = 1Hz。
另外，將 counter 中的第 16、17bit 抓出來當作給 scan control 的控制項。

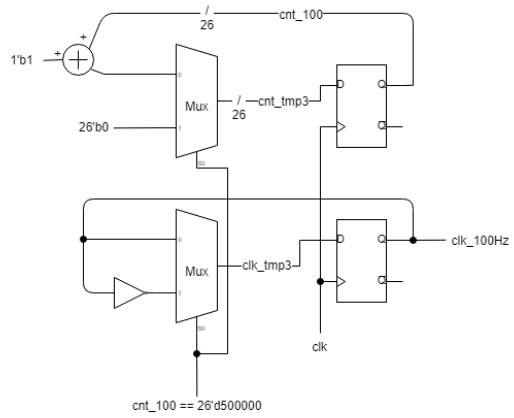
<u>10Hz:</u> 觀察秒



由於 FPGA 板上的頻率為 100MHz，建構理念是設計一個上限為 5M 的 binary up counter，在達到 5M 之前，每個 clk 都讓 counter 加上 1，而當 counter 數到 5M

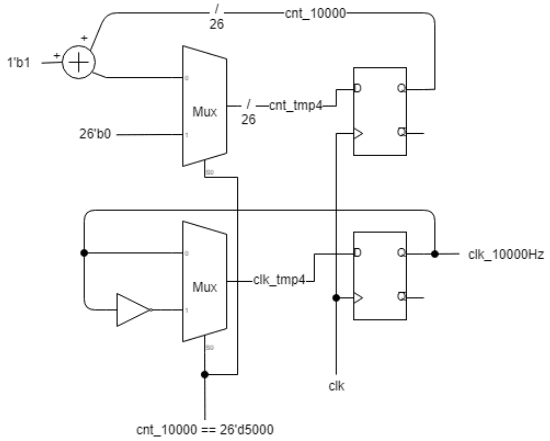
時，便將輸出值(clk_10Hz) toggle 一次，也將 counter 歸零，如此下來，輸出值(clk_10Hz)在等於 0 和等於 1 的時間各自都是 5M 次的 clk，也就是說，clk_10Hz 的值 0.05 秒會 toggle 一次，因此輸出值的頻率將會等於 100MHz/(5M*2) = 10Hz。

100Hz: 觀察分



由於 FPGA 板上的頻率為 100MHz，建構理念是設計一個上限為 500k 的 binary up counter，在達到 500k 之前，每個 clk 都讓 counter 加上 1，而當 counter 數到 500k 時，便將輸出值(clk_100Hz) toggle 一次，也將 counter 歸零，如此下來，輸出值(clk_100Hz)在等於 0 和等於 1 的時間各自都是 500k 次的 clk，也就是說，clk_100Hz 的值 0.005 秒會 toggle 一次，因此輸出值的頻率將會等於 100MHz/(500k*2) = 100Hz。

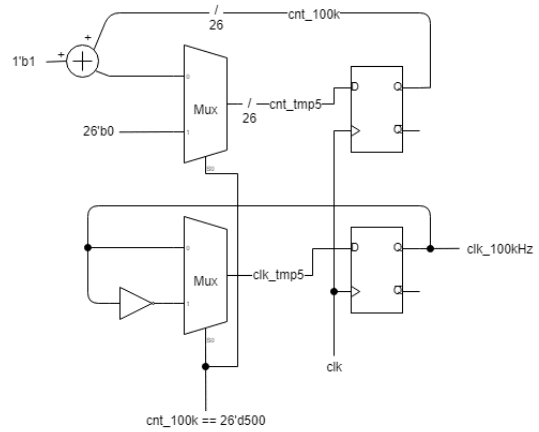
10000Hz: 觀察小時



由於 FPGA 板上的頻率為 100MHz，建構理念是設計一個上限為 5k 的 binary up counter，在達到 5k 之前，每個 clk 都讓 counter 加上 1，而當 counter 數到 5k 時，便將輸出值(clk_10000Hz) toggle 一次，也將 counter 歸零，如此下來，輸出值(clk_10000Hz)在等於 0 和等於 1 的時間各自都是 5k 次的 clk，也就是說，

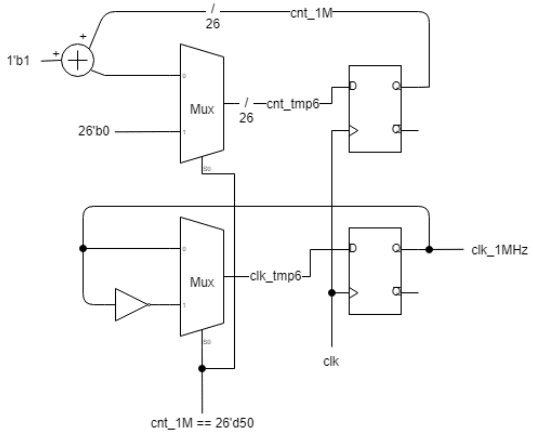
clk_10000Hz 的值 0.00005 秒會 toggle 一次，因此輸出值的頻率將會等於 100MHz/(5k*2) = 10000Hz。

100kHz: 觀察日



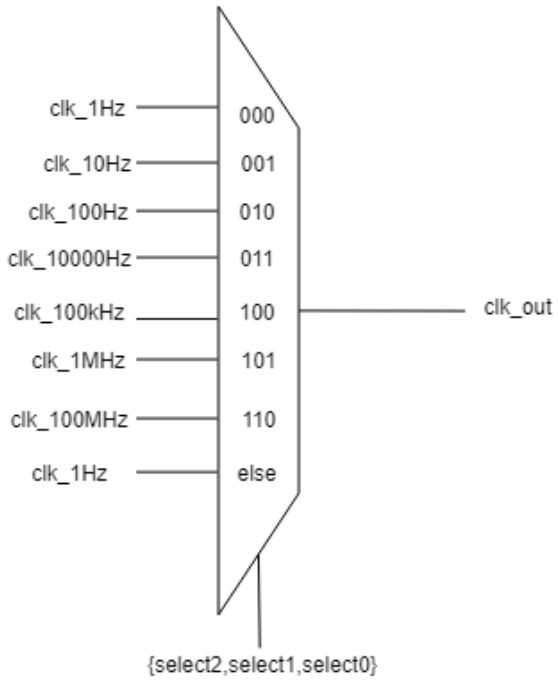
由於 FPGA 板上的頻率為 100MHz，建構理念是設計一個上限為 500 的 binary up counter，在達到 500 之前，每個 clk 都讓 counter 加上 1，而當 counter 數到 500 時，便將輸出值(clk_100kHz) toggle 一次，也將 counter 歸零，如此下來，輸出值(clk_100kHz)在等於 0 和等於 1 的時間各自都是 500 次的 clk，也就是說，clk_100kHz 的值 0.000005 秒會 toggle 一次，因此輸出值的頻率將會等於 100MHz/(500*2) = 100kHz。

1MHz: 觀察月



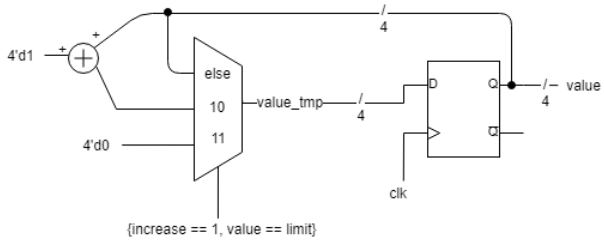
由於 FPGA 板上的頻率為 100MHz，建構理念是設計一個上限為 50 的 binary up counter，在達到 50 之前，每個 clk 都讓 counter 加上 1，而當 counter 數到 50 時，便將輸出值(clk_1MHz) toggle 一次，也將 counter 歸零，如此下來，輸出值(clk_1MHz)在等於 0 和等於 1 的時間各自都是 50 次的 clk，也就是說，clk_1MHz 的值 0.0000005 秒會 toggle 一次，因此輸出值的頻率將會等於 100MHz/(50*2) = 1MHz。

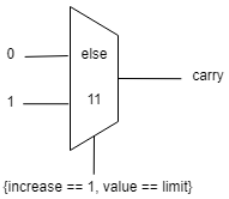
<u>頻率選擇:</u> 將除頻器輸出的 6 種頻率加上未經除頻的 100MHz 頻率(觀察年)透過三個 DIP switch(freq_sel0、freq_sel1、freq_sel2)開關選擇其中一個頻率輸出。



當三個開關為 000(預設)時 → 輸出 1Hz 頻率(clk_out = clk_1Hz) (正常)；
當三個開關為 001 時 → 輸出 10Hz 頻率(clk_out = clk_10Hz) (觀察秒)；
當三個開關為 010 時 → 輸出 100Hz 頻率(clk_out = clk_100Hz) (觀察分)；
當三個開關為 011 時 → 輸出 10000Hz 頻率(clk_out = clk_10000Hz) (觀察時)。
當三個開關為 100 時 → 輸出 100kHz 頻率(clk_out = clk_100kHz) (觀察日)；
當三個開關為 101 時 → 輸出 1MHz 頻率(clk_out = clk_1MHz)(觀察月)；
當三個開關為 110 時 → 輸出 100MHz 頻率(clk_out = clk_100MHz)(觀察年)；
當三個開關為 111 時 → 輸出 1Hz 頻率(clk_out = clk_1Hz)。

十進位上數器(秒*2、分*2、小時*2): 依據頻率選擇後得到的頻率快慢與上級的進位來進行上數。

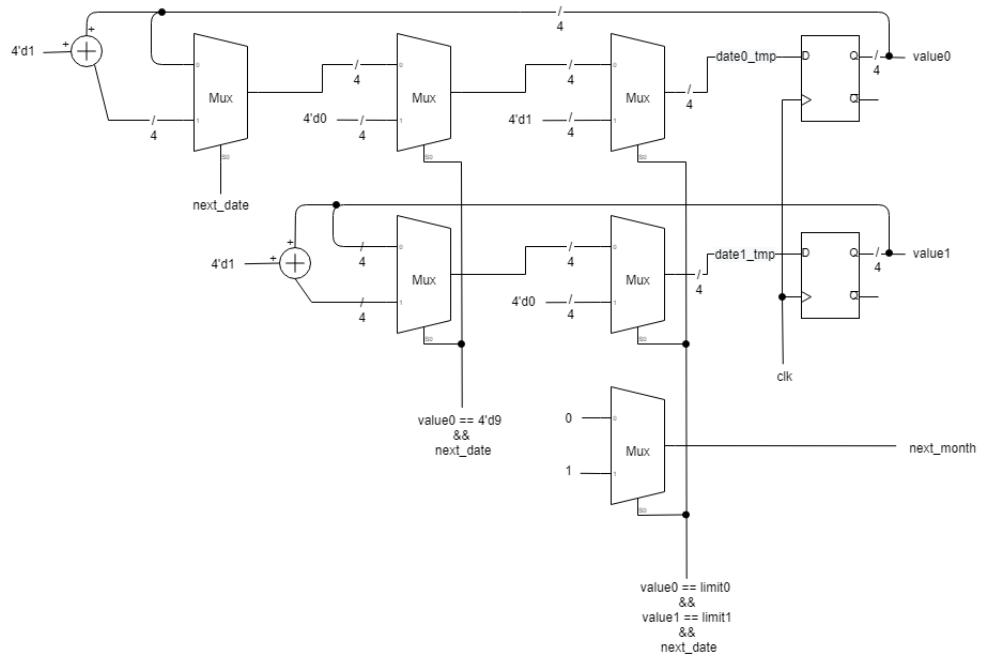




秒數個位數: (increase 為 1, limit 為 9)
秒數十位數: (increase 為秒數個位數的 carry, limit 為 5)
分鐘個位數: (increase 為秒數個位和十位數的 carry(皆為 1 時), limit 為 9)
分鐘十位數: (increase 為秒數個位、十位數以及分鐘個位數的 carry(皆為 1 時), limit 為 5)
小時個位數: (increase 為秒數個位、十位數以及分鐘個位、十位數的 carry(皆為 1 時), limit 為 9)
小時十位數: (increase 為秒數個位、十位數和分鐘個位、十位數以及小時個位數的 carry(皆為 1 時), limit 為 2)

當 increase 等於 1 且 value 等於 limit 時，代表下次需要進位，carry 輸出 1 告知下一級，value_tmp = 0；當 increase 等於 1 但 value 不等於 limit 時，代表下次要加 1 但不用進位，carry = 0，value_tmp = value + 1；其他狀況代表不需要加 1 也不需要進位，carry = 0，value 維持現狀，value_tmp = value。當每次 clk 的 positive edge 來時，將 value 輸入 value_tmp 的值。
在小時的上數器中，由於小時上限為 23(0～23)，因此另外再加上一個判斷為當小時十位數 = 2 且小時個位數 = 3 時，小時的 value_tmp 都 = 0，因此下一個 clk 時小時的 value 都會輸入 0，重新從 0 開始數。

日期上數器: 依據頻率選擇後得到的頻率快慢與上級的進位來進行上數(數到月份上數器所給的的日數上限)。

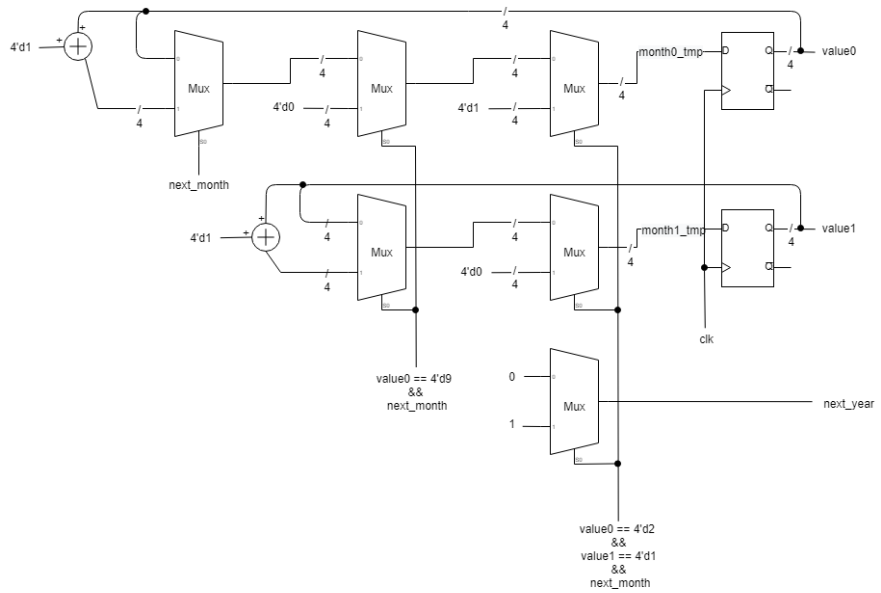


next_date 為(時、分、秒)所有上數器的 carry 都等於 1 時，也就是日期要加 1 的時候的通知信號。
value0 是日期個位數、value1 是日期十位數。
limit1、limit0 為當月日數上限的十位數與個位數。

(1)當 value0 = limit0、value1 = limit1、next_date = 1 時 → 下一天為下個月的 01 日 → date1_tmp = 0 , date0_tmp = 1。
(2)當 value0 = 9、next_date = 1 且不符合(1)時 → 個位數進位、十位數加 1 → date1_tmp = value1 + 1 , date0_tmp = 0。
(3)當 next_date = 1 且不符合(1)跟(2)時 → 個位數加 1、十位數不變 → date1_tmp = value1 , date0_tmp = value0 + 1。
(4)其餘狀況為上級未進位 → 日期不需要變動。
當每次 clk 的 positive edge 來時，將各 value 輸入各 value_tmp 的值。

當 value0 = limit0、value1 = limit1、next_date = 1 時 → 下一天為下個月的 01 日 → 告知月份上數器要加 1 → next_month = 1；其餘狀況月份不需要加 1 → next_month = 0。

<u>月份上數器:</u> 依據頻率選擇後得到的頻率快慢與上級的進位來進行上數；另外，判斷當前的月份得到當月的總日數，並告知日期上數器。

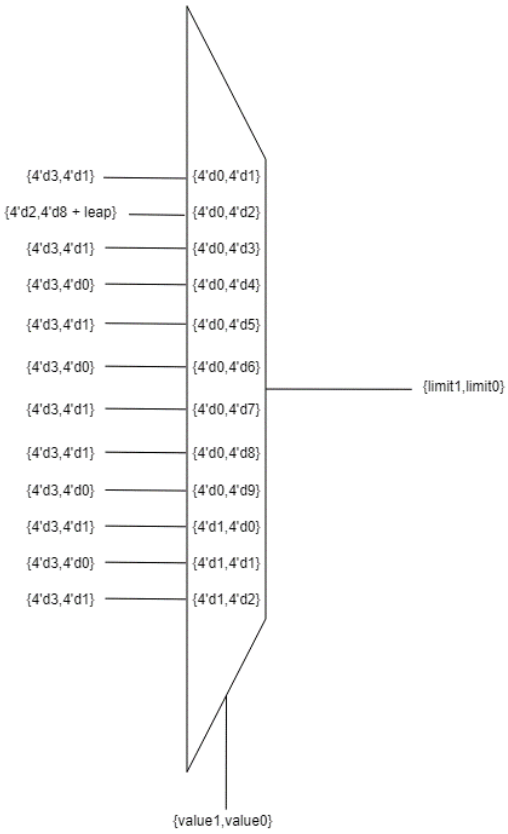


next_month 為(時、分、秒、日)所有上數器的 carry 都等於 1 時，也就是月份要加 1 的時候的通知信號。
value0 是月份個位數、value1 是月份十位數。

(1)當 value0 = 2、value1 = 1、next_month = 1 時 → 下個月為隔年 01 月→ date1_tmp = 0 , date0_tmp = 1。
(2)當 value0 = 9、next_month = 1 且不符合(1)時 → 個位數進位、十位數加 1 → date1_tmp = value1 + 1 , date0_tmp = 0。
(3)當 next_month = 1 且不符合(1)跟(2)時 → 個位數加 1、十位數不變 → date1_tmp = value1 , date0_tmp = value0 + 1。
(4)其餘狀況為上級未進位 → 月份不需要變動。
當每次 clk 的 positive edge 來時，將各 value 輸入各 value_tmp 的值。

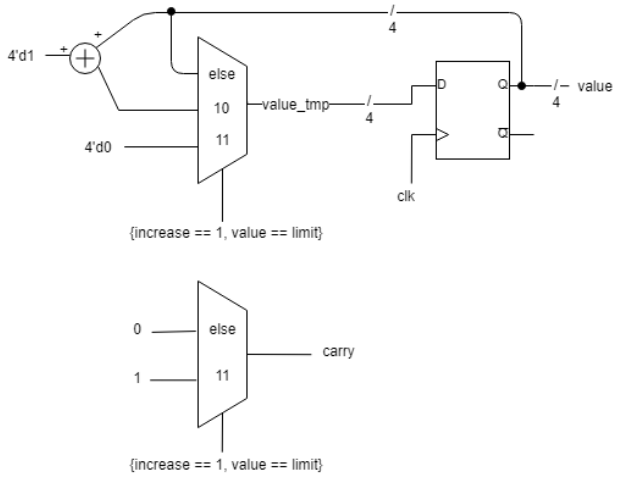
當 value0 = 2、value1 = 1、next_month = 1 時 → 下個月為隔年 01 月 → 告知年份上數器要加 1 → next_year = 1；其餘狀況年份不需要加 1 → next_year = 0。

limit1、limit0 為當月日數上限的十位數與個位數。
leap 為閏年檢查得到的結果: 是閏年→1；不是閏年→0。



(1) value1 = 0、value0 = 1 → 1 月 → 共 31 天 → limit1 = 3, limit0 = 1
(2) value1 = 0、value0 = 2 → 2 月 → 共 28 天 → limit1 = 2, limit0 = 8 + leap
(3) value1 = 0、value0 = 3 → 3 月 → 共 31 天 → limit1 = 3, limit0 = 1
(4) value1 = 0、value0 = 4 → 4 月 → 共 30 天 → limit1 = 3, limit0 = 0
(5) value1 = 0、value0 = 5 → 5 月 → 共 31 天 → limit1 = 3, limit0 = 1
(6) value1 = 0、value0 = 6 → 6 月 → 共 30 天 → limit1 = 3, limit0 = 0
(7) value1 = 0、value0 = 7 → 7 月 → 共 31 天 → limit1 = 3, limit0 = 1
(8) value1 = 0、value0 = 8 → 8 月 → 共 31 天 → limit1 = 3, limit0 = 1
(9) value1 = 0、value0 = 9 → 9 月 → 共 30 天 → limit1 = 3, limit0 = 0
(10) value1 = 1、value0 = 0 → 10 月 → 共 31 天 → limit1 = 3, limit0 = 1
(11) value1 = 1、value0 = 1 → 11 月 → 共 30 天 → limit1 = 3, limit0 = 0
(12) value1 = 1、value0 = 2 → 12 月 → 共 31 天 → limit1 = 3, limit0 = 1

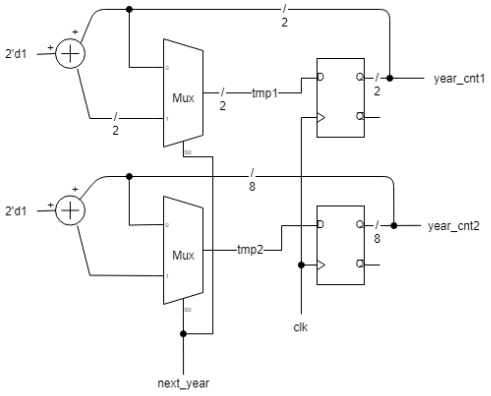
年份上數器*3: 依據頻率選擇後得到的頻率快慢與上級的進位來進行上數。



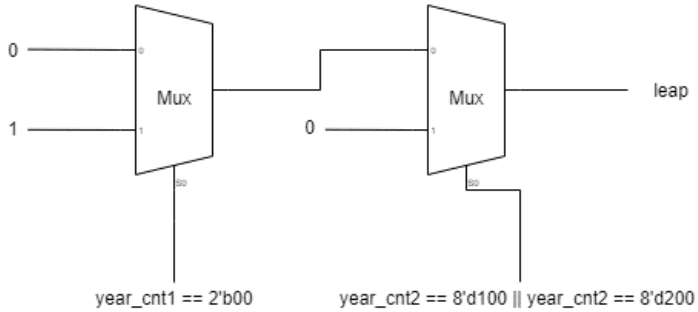
年份個位數: (increase 為月份要進位的信號 next_year, limit 為 9)
年份十位數: (increase 年份個位數的 carry, limit 為 9)
年份百位數: (increase 年份個位數和十位數的 carry 皆為 1 時, limit 為 2)

當 increase 等於 1 且 value 等於 limit 時，代表下次需要進位，carry 輸出 1 告知下一級，value_tmp = 0；當 increase 等於 1 但 value 不等於 limit 時，代表下次要加 1 但不用進位，carry = 0，value_tmp = value + 1；其他狀況代表不需要加 1 也不需要進位，carry = 0，value 維持現狀，value_tmp = value。當每次 clk 的 positive edge 來時，將 value 輸入 value_tmp 的值。

閏年檢查: 連接通知年份進位的信號(next_year)當輸入，並計算該年是否為閏年，並將結果告知月份上數器中判斷該月總日數的地方。



建造一個 2bit 的 counter 和一個 8bit 的 counter，當通知年份進位的信號(next_year)為 1 時，讓兩個 counter 各自都加 1；當通知年份進位的信號(next_year)為 0 時，兩個 counter 各自等於原本的值。
也就是說，每過一年就讓 counter 加 1。



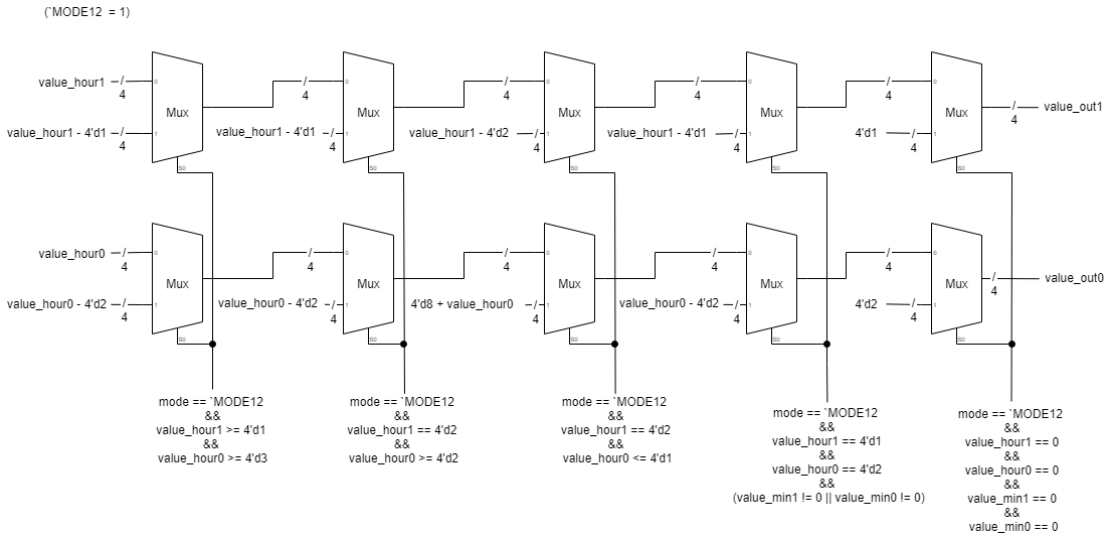
2bit 的 counter 初始值為 2’b00，由於是從 2000 年開始(是閏年)，因此每當數到 00 時只要當年年份不是 100 的倍數就是閏年。

(1)當 year_cnt2 == 8'd100 或 year_cnt2 == 8'd200 → 不是閏年 → leap = 0；

(2)當 year_cnt1 == 2'b00 且不符合(1) → 是閏年 → leap = 1；

(3)其餘狀況皆不是閏年 → leap = 0。

<u>mode 選擇:</u> 在預設狀況下為 24 小時制，若將開關(modeAMPM)打開，則變換成 12 小時制。



以 mode 連接開關(modeAMPM)的訊號。

value_hour1: 24 小時制小時十位數輸入值

value_hour0: 24 小時制小時個位數輸入值

value_out1: 小時十位數輸出值

value_out0: 小時個位數輸出值

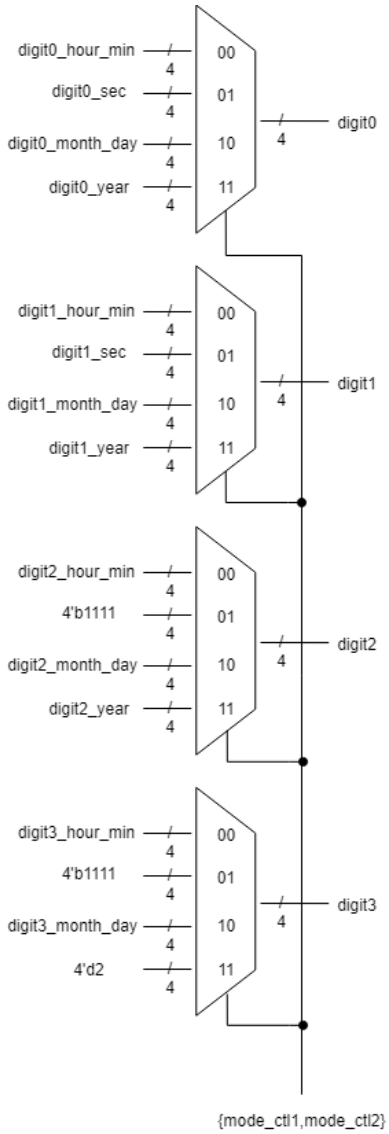
(1)當 mode = 1、小時為 00、分鐘為 00 → 現在為午夜 00:00(AM12:00)且需要轉換成 12 小時制 → 十位數 = 1, 個位數 = 2。

(2)當 mode = 1、小時為 12、分鐘不為 00 且不符合(1)時 → 現在為中午 12:01～12:59(PM00:01～00:59)且需要轉換成 12 小時制 → 十位數減 1, 個位數減 2。

(3)當 mode = 1、小時為 20～21 且不符合(1)跟(2)時 → 現在為晚上 20:00～21:59(PM08:00～09:59)且需要轉換成 12 小時制但個位數不夠減 → 十位數減 2, 個位數加 8。
(4)當 mode = 1、小時為 22～23 且不符合(1)～(3)時 → 現在為晚上 22:00～23:59(PM10:00～11:59)且需要轉換成 12 小時制且個位數夠減 → 十位數減 1, 個位數減 2。
(5)當 mode = 1、小時為 13～19 且不符合(1)～(4)時 → 現在為 13:00～19:59(PM01:00～07:59)且需要轉換成 12 小時制 → 十位數減 1, 個位數減 2。

經過以上(1)～(5)之後，所有需要轉換的部分都已經處理完畢，因此
(6)其餘狀況: 輸出值等於輸入值。

MUX: 透過兩個開關(mode_ctl1、mode_ctl2)來控制要顯示「小時-分鐘」或是顯示「秒鐘」或是顯示「月-日」或是顯示「年」，預設情形為顯示「小時-分鐘」。



將 digit3～2_hour_min 連接 mode 的輸出值(小時)；
將 digit1～0_hour_min 連接分鐘上數器的輸出值(分鐘)；
將 digit1～0_sec 連接秒數上數器的輸出值(秒數)；
將 digit3～2_ month_day 連接月份上數器的輸出值；
將 digit1～0_ month_day 連接日期上數器的輸出值；
將 digit2～0_year 連接年份上數器的輸出值。

當{mode_ctl1, mode_ctl2} == 2'b00 → 顯示「小時-分鐘」 → digit3～0 輸入 digit3～0_hour_min；
當{mode_ctl1, mode_ctl2} == 2'b01 → 顯示「秒鐘」 → digit3～2 輸入 4'b1111, digit1～0 輸入 digit1～0_sec；(只顯示兩位數)
當{mode_ctl1, mode_ctl2} == 2'b10 → 顯示「月-日」 → digit3～0 輸入 digit3～0_month_day；
當{mode_ctl1, mode_ctl2} == 2'b11 → 顯示「年」 → digit3 輸入 4'd2(因為是從 2000 年到 2200 年，千位數不變), digit2～0 輸入 digit2～0_ year。

Scan control: 將 MUX 的輸出 digit3～0 以及除頻器的輸出 ssd control enable 作為輸入，透過快速的輪流顯示，可以讓 4 個七段顯示器各自顯示自己要顯示的數字，並讓眼睛看起來是 4 個顯示器同時顯示的。

ssd_ctl_en = 00 → ssd_ctl = 0111
ssd_ctl_en = 01 → ssd_ctl = 1011
ssd_ctl_en = 10 → ssd_ctl = 1101
ssd_ctl_en = 11 → ssd_ctl = 1110

四個七段顯示器輪流顯示，ssd_ctl_en = 00 時→顯示第一個七段顯示器(依據模式顯示 1.小時十位數 2.不亮 3.月份十位數 4.年份千位數)；ssd_ctl_en = 01 時→顯示第二個七段顯示器(依據模式顯示 1.小時個位數 2.不亮 3.月份個位數 4.年份百位數)；ssd_ctl_en = 10 時→顯示第三個七段顯示器(依據模式顯示 1.分鐘十位數 2.秒數十位數 3.日期十位數 4.年份十位數)；ssd_ctl_en = 11 時→顯示第四個七段顯示器(依據模式顯示 1.分鐘個位數 2.秒數個位數 3.日期個位數 4.年份個位數)，以快速輪流顯示的方式達成視覺暫留的效果。

7-segment display decoder: 將 scan control 給的輸入值(ssd_in)透過解碼之後顯示在七段顯示器上。

in = 0 → segs = 8'b0000_0011
in = 1 → segs = 8'b1001_1111

in = 2 → segs = 8'b0010_0101
in = 3 → segs = 8'b0000_1101
in = 4 → segs = 8'b1001_1001
in = 5 → segs = 8'b0100_1001
in = 6 → segs = 8'b0100_0001
in = 7 → segs = 8'b0001_1111
in = 8 → segs = 8'b0000_0001
in = 9 → segs = 8'b0000_1001
default 為 segs = 8'b1111_1111(全暗)

結合以上功能，可以完成一個支援閏年、可以切換顯示「小時-分鐘」/「秒鐘」/「月-日」/「年」、可以切換 24 小時制/12 小時制的電子鐘。

I/O pin assignment:

segs[7]	segs[6]	segs[5]	segs[4]	segs[3]	segs[2]	segs[1]	segs[0]
W7	W6	U8	V8	U5	V5	U7	V7

ssd_ctl[3]	ssd_ctl[2]	ssd_ctl[1]	ssd_ctl[0]	clk	rst_n
W4	V4	U4	U2	W5	R2

mode_ctl1	mode_ctl2	modeAMPM	freq_sel0	freq_sel1	freq_sel2
W16	V16	V17	W2	U1	T1

Conclusion:

這次的實驗將之前所學的東西結合起來並實際運用在一個系統內，如多個 BCD up counter 連動的應用，題目整題而言難度雖然不高，但是卻要考慮到非常多小細節，像是 counter 每個位數的上限值、數到哪個值要重置、閏年的計算、12/24 小時制的變換問題等等。這些細節都讓這個看似簡單的題目變得非常複雜，感覺上反而像是在打 C 語言，不過也許實際應用上的電路都並不是像之前那樣那麼理想化，需要考慮到的情況可能也會越來越多，如何妥善運用之前完成的 module 也變成了重要的課題。