



OPL1000 Peripheral PWM Application Notes

SALES@OPULINKS.COM

DEC 26, 2018





Revision History

| 版本号 | 时间 | 说明 |
|------|------------|---|
| v0.1 | 2018/05/23 | Draft version |
| v0.2 | 2018/05/31 | Update according to TW team review feedback |
| v0.3 | 2018/12/26 | Updated for A2 board, update page5, 9 and 23. |
| v0.4 | 2019/12/26 | Updated for PWM example complex mode, add page 22 and 23. |



- PWM Module features
- PWM Port Resource and Setting
- Configuration by Pin-Mux Tool
- PWM module API
- Simple Configure Mode with different clock source
- Square Wave use complex configure mode
- Complex Configuration Mode Example
- Application Notice
- Reference



PWM Module Features

- Electrical spec: up to VDDO, $V_{OL} = 0.4v$, $V_{OH} = 2.4v$; $I_{OL} = 29.5mA$ (typical), $I_{OH} = 60.3mA$ (typical)
- Two clock source: 32KHz and 22MHz clock
- Two configuration mode: Simple mode and Complex mode.
- Simple Mode: 3 parameters need to set.
 - ① Clock source
 - ② Duty rate : 1%~100%, precise 1%
 - ③ Output frequency (Hz unit)
- Complex Mode: 7 parameters, duty rate can varies, precise is higher than simple mode.
 - ① PWM period, defined in clock cycles number
 - ② Ramp up and ramp down interval
 - ③ Bright and Dull duration can be defined independently
 - ④ Hold bright and dull duration can be defined independently

PWM Port Resource and Setting

OPL1000 support 6 PWM port, PWM0~PWM5

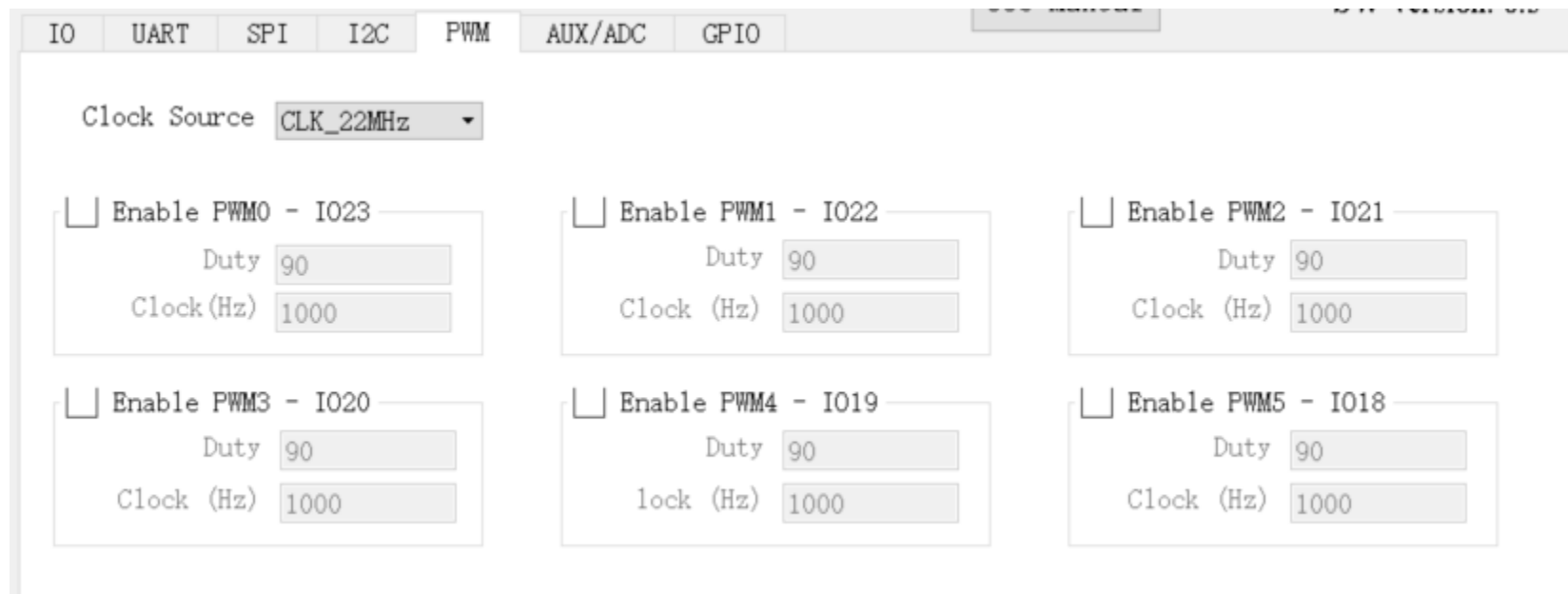
| Index | PWM | IO-Idx | Pin-Loc | Pin @DEVKIT |
|-------|------|--------|---------|-------------|
| 1 | PWM0 | IO23 | 29 | J3 – Pin3 |
| 2 | PWM1 | IO22 | 48 | J2 – Pin16 |
| 3 | PWM2 | IO21 | 47 | J2 – Pin14 |
| 4 | PWM3 | IO20 | 46 | J2 – Pin13 |
| 5 | PWM4 | IO19 | 45 | J2 – Pin12 |
| 6 | PWM5 | IO18 | 44 | J2 – Pin11 |

DEVKIT GPIO mapping

| J2 | | | | | | ANT | J3 | | | | | |
|----------|-----|------|-----|----------|--------|-------------|--------|------------|-----|------|------|-------------|
| ICE Mode | PWM | I2C | ADC | Pin Name | Pin No | | Pin No | Pin Name | ADC | SPI | UART | Flash Prg |
| | | | | GND | pin 17 | Bottom View | pin 17 | GND | | | | |
| | Yes | | | GPIO22 | pin 16 | | pin 16 | +3V | | | | |
| | | | | GND | pin 15 | | pin 15 | GND | | | | |
| M3_CLK | | | | GPIO21 | pin 14 | | pin 14 | CHIP_EN | | | | |
| M3_DAT | | | | GPIO20 | pin 13 | | pin 13 | RST_N | | | | |
| M0_DAT | | | | GPIO19 | pin 12 | | pin 12 | GPIO0(REV) | | | | UART_Prg_Tx |
| M0_CLK | | | | GPIO18 | pin 11 | | pin 11 | GPIO1(REV) | | | | UART_Prg_Rx |
| | | SDA | Yes | GPIO17 | pin 10 | | pin 10 | GPIO2 | Yes | MOSI | TxD | |
| | | SCLK | Yes | GPIO16 | pin 9 | | pin 9 | GPIO3 | Yes | MISO | RxD | |
| | | | | GPIO15 | pin 8 | | pin 8 | GPIO4 | Yes | CLK | | |
| | | | | GPIO14 | pin 7 | | pin 7 | Ex_5V | | | | |
| | | | | GPIO13 | pin 6 | | pin 6 | GND | | | | |
| | Yes | | | GPIO12 | pin 5 | | pin 5 | GPIO5 | Yes | CS | | |
| | | | | GPIO11 | pin 4 | | pin 4 | GPIO6 | Yes | | | |
| | | | | GPIO10 | pin 3 | | pin 3 | GPIO23 | | | | |
| | Yes | | | GPIO9 | pin 2 | | pin 2 | GPIO7 | Yes | CS | | |
| | | | | GND | pin 1 | USB | pin 1 | GPIO8 | Yes | | | |

Configuration by Pin-Mux Tool

- Pin-Mux tool only provide simple mode PWM configuration method
- Complex mode setting need to define PWM module of OPL1000_periph directly
- Clock source shall be same for multiple PWM port



The screenshot shows the PWM configuration tab in the Pin-Mux Tool. At the top, there are tabs for IO, UART, SPI, I2C, PWM, AUX/ADC, and GPIO. The PWM tab is selected. Below the tabs, the 'Clock Source' is set to 'CLK_22MHz'. There are six PWM channels, each with an 'Enable' checkbox, a 'Duty' input field, and a 'Clock (Hz)' input field. All channels are currently disabled, and their duty and clock values are set to 90 and 1000 respectively.

| Channel | Enable | Duty | Clock (Hz) |
|-------------|--------------------------|------|------------|
| PWM0 - IO23 | <input type="checkbox"/> | 90 | 1000 |
| PWM1 - IO22 | <input type="checkbox"/> | 90 | 1000 |
| PWM2 - IO21 | <input type="checkbox"/> | 90 | 1000 |
| PWM3 - IO20 | <input type="checkbox"/> | 90 | 1000 |
| PWM4 - IO19 | <input type="checkbox"/> | 90 | 1000 |
| PWM5 - IO18 | <input type="checkbox"/> | 90 | 1000 |



PWM Module API (1)

- Hal_Pinmux_Pwm_Init: Initialize PWM module
- Hal_Pinmux_Pwm_Config: Configure PWM port, include pin assignment and parameter setting
- Hal_Pinmux_Pwm_Enable: Enable defined PWM port, single or multiple
- Hal_Pinmux_Pwm_Disable: Disable PWM port, certain on port or multiple ports
- Hal_PinMux_Get_Index : Get PWM index according to assigned Pin number
- T_OPL1000_Periph OPL1000_periph: global variable, defines peripheral resource, include PWMport
 - Element: uint8_t pwm_num – defines how many PWM ports need to config
 - Element: _OPL1000_Pwm pwm[PWM_MAX_NUM] – defined pin assignment and other parameters





PWM Module API (2)

T_OPL1000_Pwm structure definition:

- uint8_t pin; // PWM pin assignment
- E_PwmClkSrc_t clkSrc; // clock source, 32kHz or 22MHz clock
- E_PwmCfgType_t cfgType; // Configuration mode, simple or complex mode
- uint8_t duty; // duty rate, from 1 to 100, corresponding to 1% to 100%
- uint32_t clkHz; // output PWM waveform frequency, in Hz unit
- uint32_t period; // the tick count in one PWM cycle
- uint32_t dutyBright; // max tick count of high level in on PWM cycle
- uint32_t dutyDull; // min tick count of high level in one PWM cycle
- uint32_t rampUp; // delta count from dull to bright per clock cycle
- uint32_t rampDown; // delta count from bright to dull per clock cycle
- uint32_t holdBright; // hold times of the bright state
- uint32_t holdDull; // hold times the dull state



PWM Module API (3)

PWM Setting process flow:

1. Use pin-mux tool to define PWM port resource (generate OPL1000_pin_mux_define.c file)
2. Get PWM port number from OPL1000_periph variable
3. Initialize PWM module
4. Disable all PWM port output
5. Enter loop processing.
 - For each PWM port , get PWM index according to IO Pin number
 - Calculate pwm_index_mask for multiple PWM port case
 - Call Hal_Pinmux_Pwm_Config to config each PWM port
5. Enable PWM port according to combined pwm_index_mask

// Example code:

```
void App_Pin_InitConfig(void)
{
    uint8_t pwm_num = OPL1000_periph.pwm_num;
    uint8_t i, pwm_index_mask = 0, pwm_idx;

    if(pwm_num > 0)
    {
        Hal_Pinmux_Pwm_Init();

        // Disable all PWM output
        Hal_Pinmux_Pwm_Disable(HAL_PWM_IDX_ALL);

        for (i=0; i<pwm_num;i++)
        {
            pwm_idx = Hal_PinMux_Get_Index(OPL1000_periph.pwm[i].pin);
            pwm_index_mask = pwm_index_mask | pwm_idx;

            Hal_Pinmux_Pwm_Config(&OPL1000_periph.pwm[i]);
        }

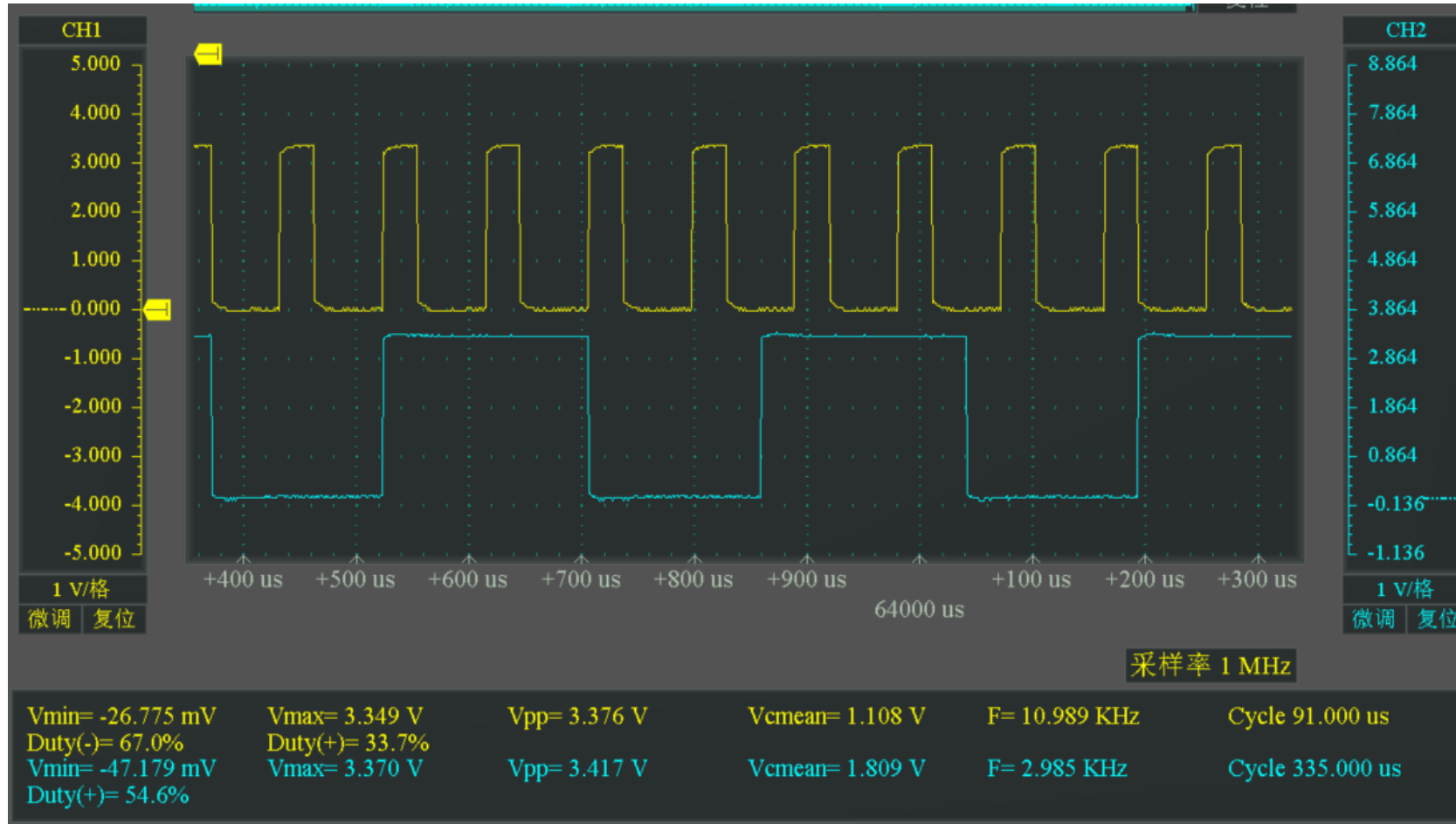
        Hal_Pinmux_Pwm_Enable(pwm_index_mask);
    }
}
```



Simple Configure Mode (32kHz clock source)

PWM1: {OPL1000_IO18_PIN, CLK_32KHz,CFG_SIMPLE,20,10000,0,0,0,0,0,0}: CH1, 20% duty, 10kHz; 32kHz clock source

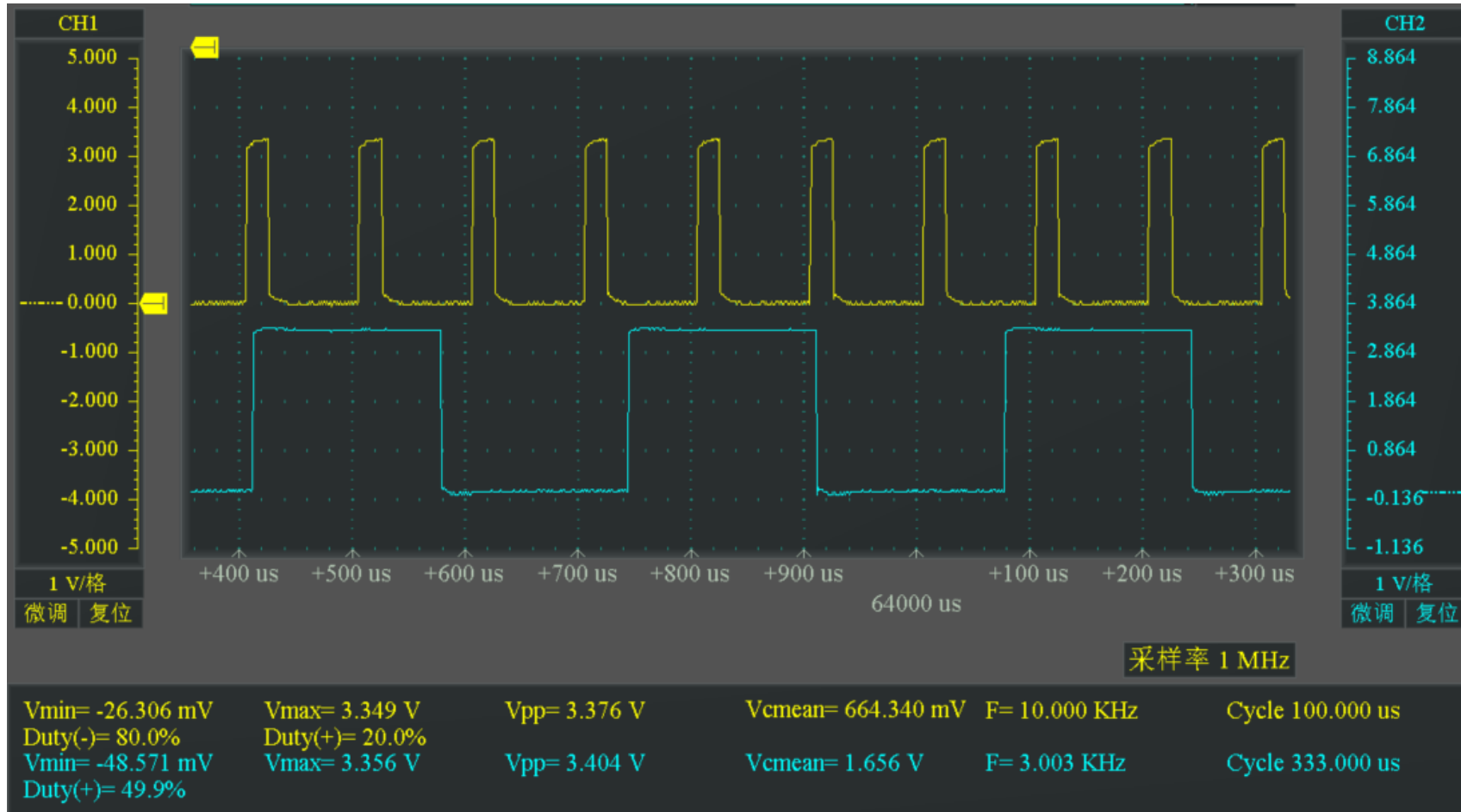
PWM4: {OPL1000_IO19_PIN, CLK_32KHz ,CFG_SIMPLE,50,3000,0,0,0,0,0,0,0}: CH2, 50% duty, 3kHz; 32kHz clock source



Simple Configure Mode (22MHz clock source)

PWM1: {OPL1000_IO18_PIN,CLK_22MHz,CFG_SIMPLE,20,10000,0,0,0,0,0,0}: CH1, 20% duty, 10kHz; 22MHz clock source

PWM4: {OPL1000_IO19_PIN,CLK_22MHz,CFG_SIMPLE,50,3000,0,0,0,0,0,0,0}: CH2, 50% duty, 3kHz; 22MHz clock source



Square Wave use complex configure mode (1)

PWM parameter setting:

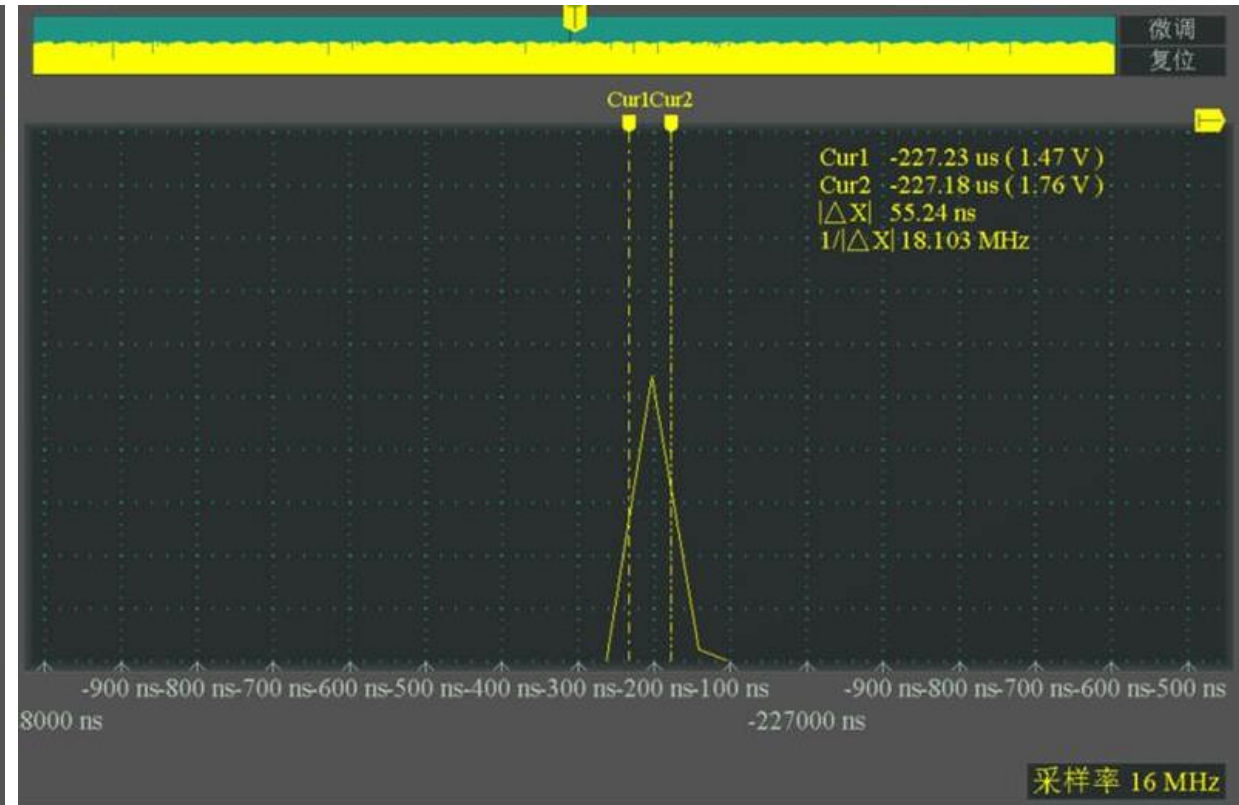
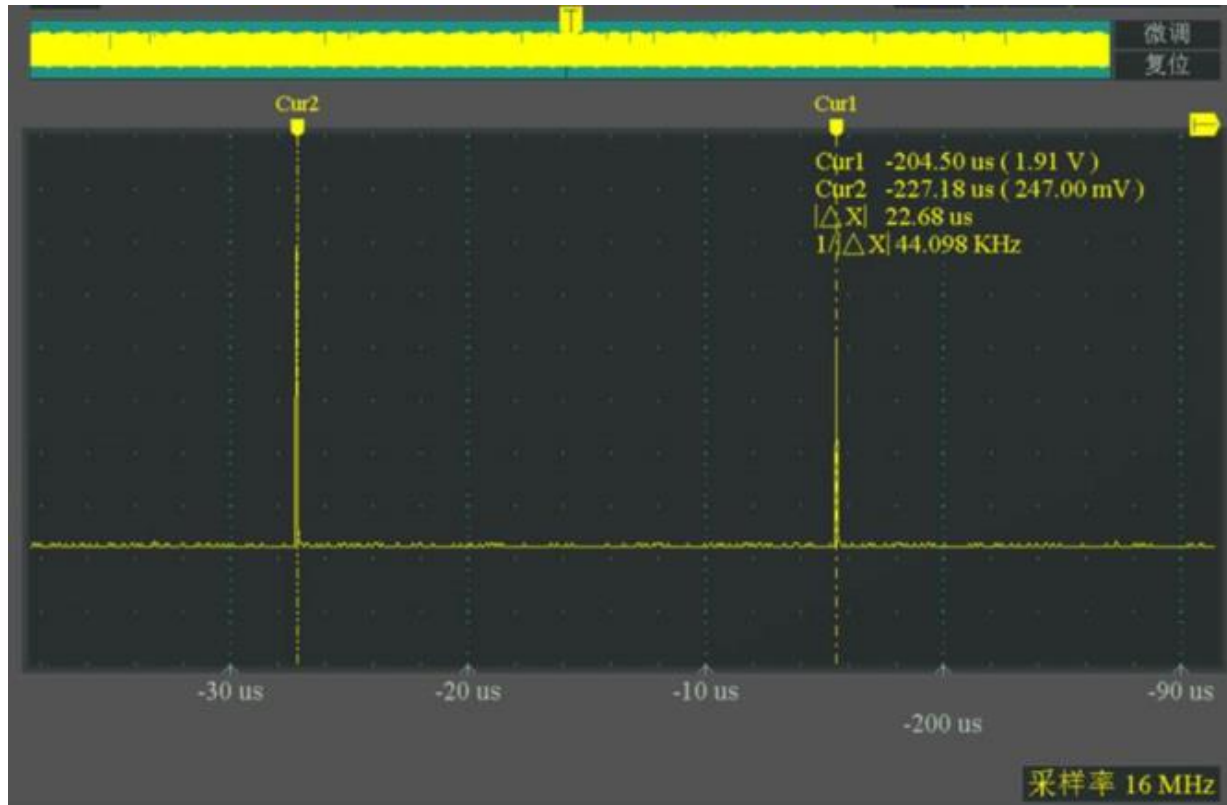
{OPL1000_IO18_PIN,CLK_22MHz,CFG_COMPLEX,20,10000,1,1,0,1,1,1,499}

- Clock cycle is $1/22\text{MHz} = 45\text{ns}$, PWM cycle = $(1+499)*\text{clock_cycle} = 22.7\mu\text{s}$
- Bright pulse (high level) width = 1 clock cycle
- Because Duty_dull = 0 and ramp_down = 1, then next cycle bright (high level) is 0
- Enter into Dull state, keep for 499 clock cycles.
- Turn to ramp up stage, duty is changed to “Bright” state, keep 1 cycle (hold_bright =1)
- Hence one repeat pattern duration is: $1+499=500$ clock cycle
- duty rate is fixed to $(1/500) = 0.2\%$. Dull rate = $1-\text{duty} = 99.8\%$

| Parameters | Value |
|-------------|-------|
| PWM period | 1 |
| Duty_Bright | 1 |
| Duty_Dull | 0 |
| Ramp_Up | 1 |
| Ramp_Down | 1 |
| Hold_Bright | 1 |
| Hold_Dull | 499 |

Complex Configuration Mode (2)

- Period = 22.68us, Freq = 1/Period = 44.1 KHz
- high level pulse width = 55.24ns, whole period = 22.68us, hence duty_rate = 0.24%



Complex Configuration Example

{OPL1000_IO18_PIN,CLK_32KHz,CFG_COMPLEX,20,10000,100,80,20,5,10,4,8}

- PWM period is fixed, but duty rate is changed between 20% ~ 80%
- Clock cycle is $1/32\text{KHz} = 31.25\mu\text{s}$, PWM cycle = $100 \times \text{clock_cycle} = 3.125\text{ms}$
- Duty is reduced from 80 clock cycle to 20, reduce interval is 10. hence needs 6 PWM cycle
- then keep Duty@20 clk_cycle for 8 PWM cycle
- Duty is increased from 20 clock cycle to 80, increase interval is 5, hence needs 12 PWM cycle
- then keep Duty@80 clk_cycle for 4 PWM cycle
- One repeat pattern duration is : $6 + 8 + 12 + 4 = 30$ PWM cycles

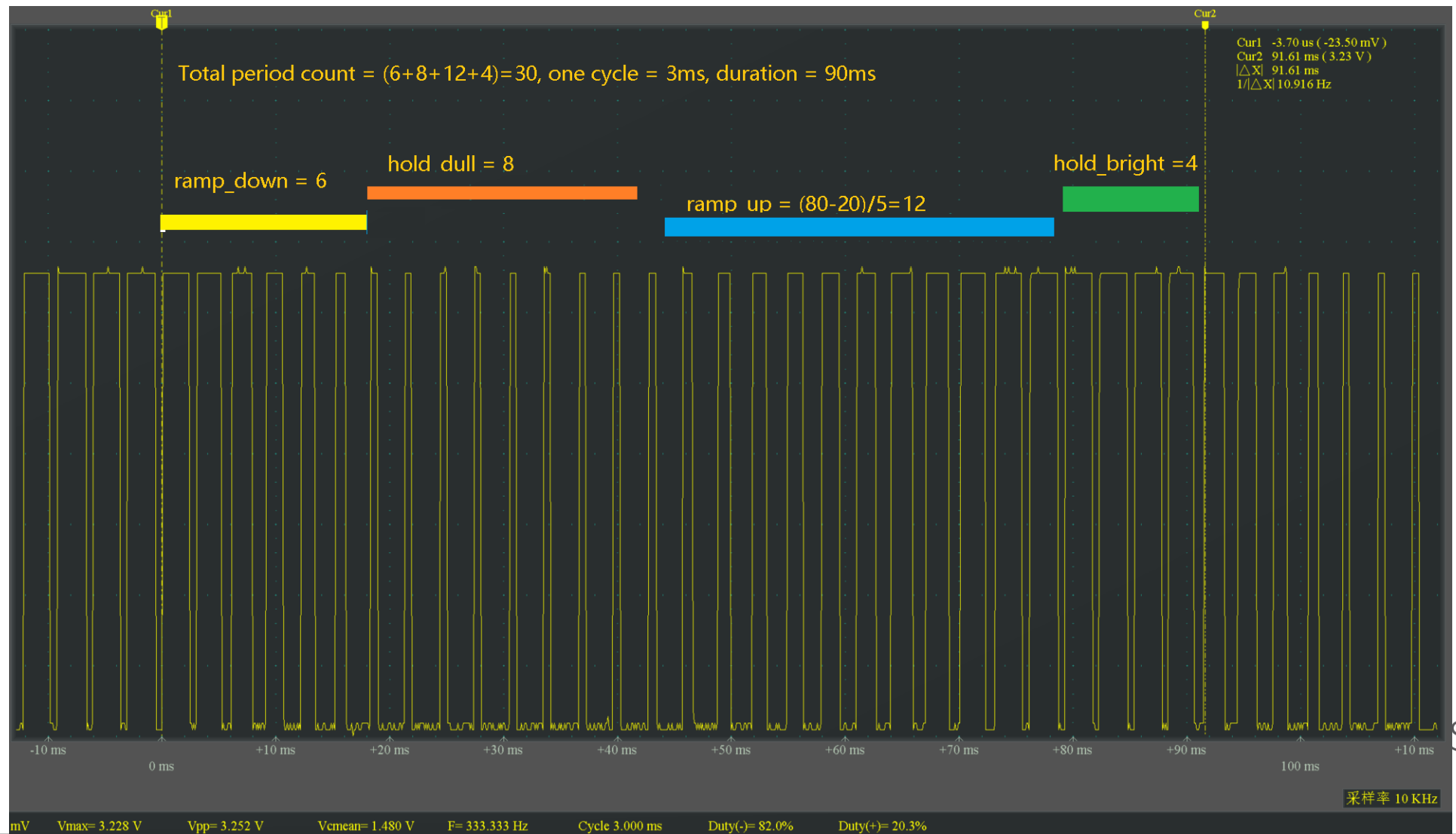
| Parameters | Value |
|-------------|-------|
| PWM period | 100 |
| Duty_Bright | 80 |
| Duty_Dull | 20 |
| Ramp_Up | 5 |
| Ramp_Down | 10 |
| Hold_Bright | 4 |
| Hold_Dull | 8 |

| Ramp_Down Duration | Hold_Dull Duration | Ramp_Up Duration | Hold_Bright Duration |
|--------------------|--------------------|------------------|----------------------|
| 6 PWM cycle | 8 PWM cycle | 12 PWM cycle | 4 PWM cycle |



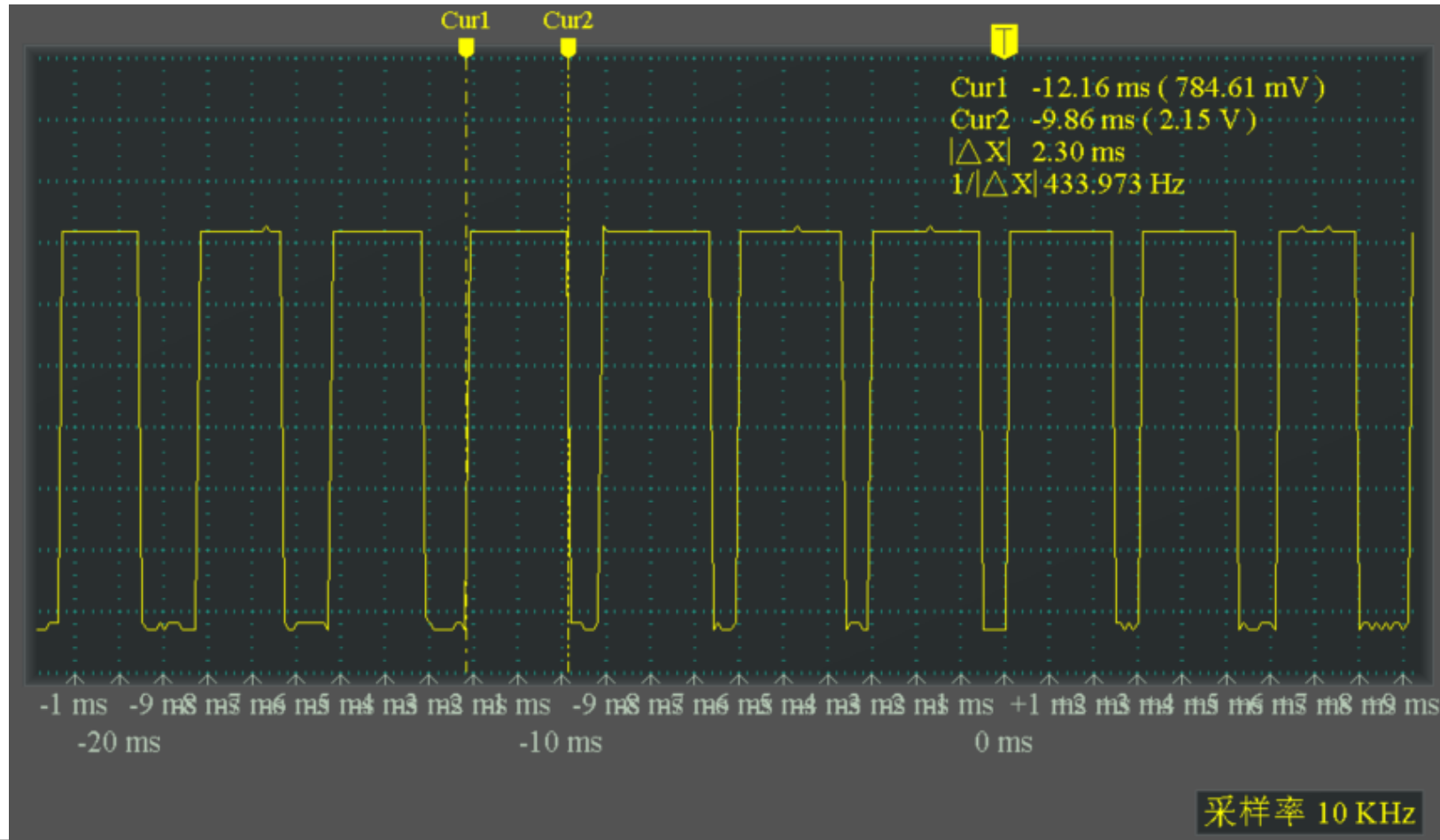
Complex Setting: Overall picture

- Ramp_up time = $(80-20)/10 = 6$
- Hold_dull = 8
- Ramp_down time = $(80-20)/5 = 12$
- Hold_Bright = 4
- Total = $6+8+12+4=30$



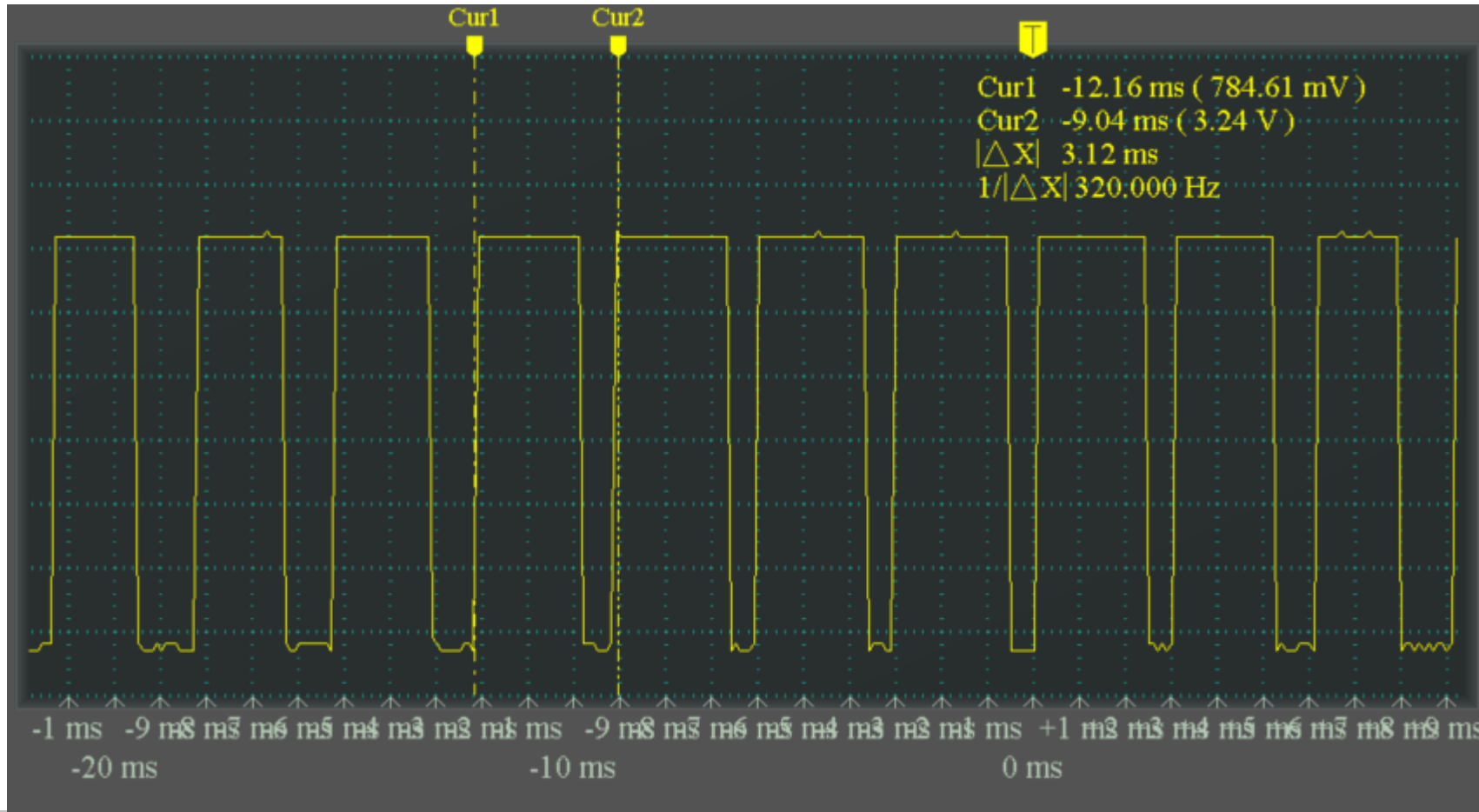
Complex Setting: Duty_Bright

- Duty_Bright = 80 cycle. In theory it is $T = 80 * (1/32\text{kHz}) = 2.5\text{ms}$
- Due to equipment measurement error, one clock cycle is 30 us instead of 31.25us, hence “bright” width = $80 * 30 = 2.4\text{ms}$



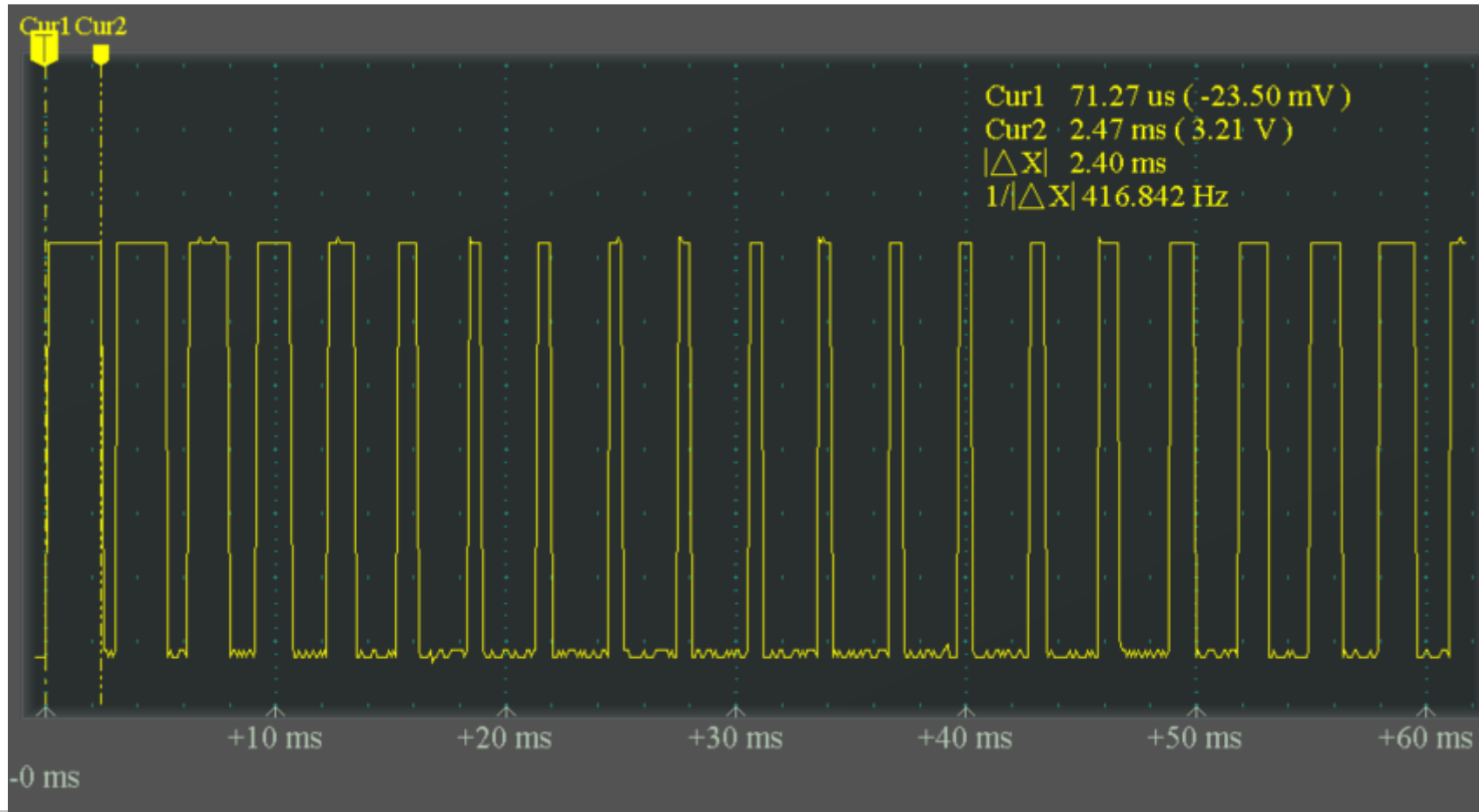
Complex Setting: Hold Bright State

- Hold_Bright = 4, back count from 0ms, there are 4 cycles BIGHT state. High pulse width = $80 * \text{clock_cycle} = 80 * (1/32\text{kHz}) = 2.5 \text{ ms}$
- From 0ms time line, Bright is 80 cycle, period = 100 cycle = $100 * (1/32\text{KHz}) = 3.125 \text{ ms}$



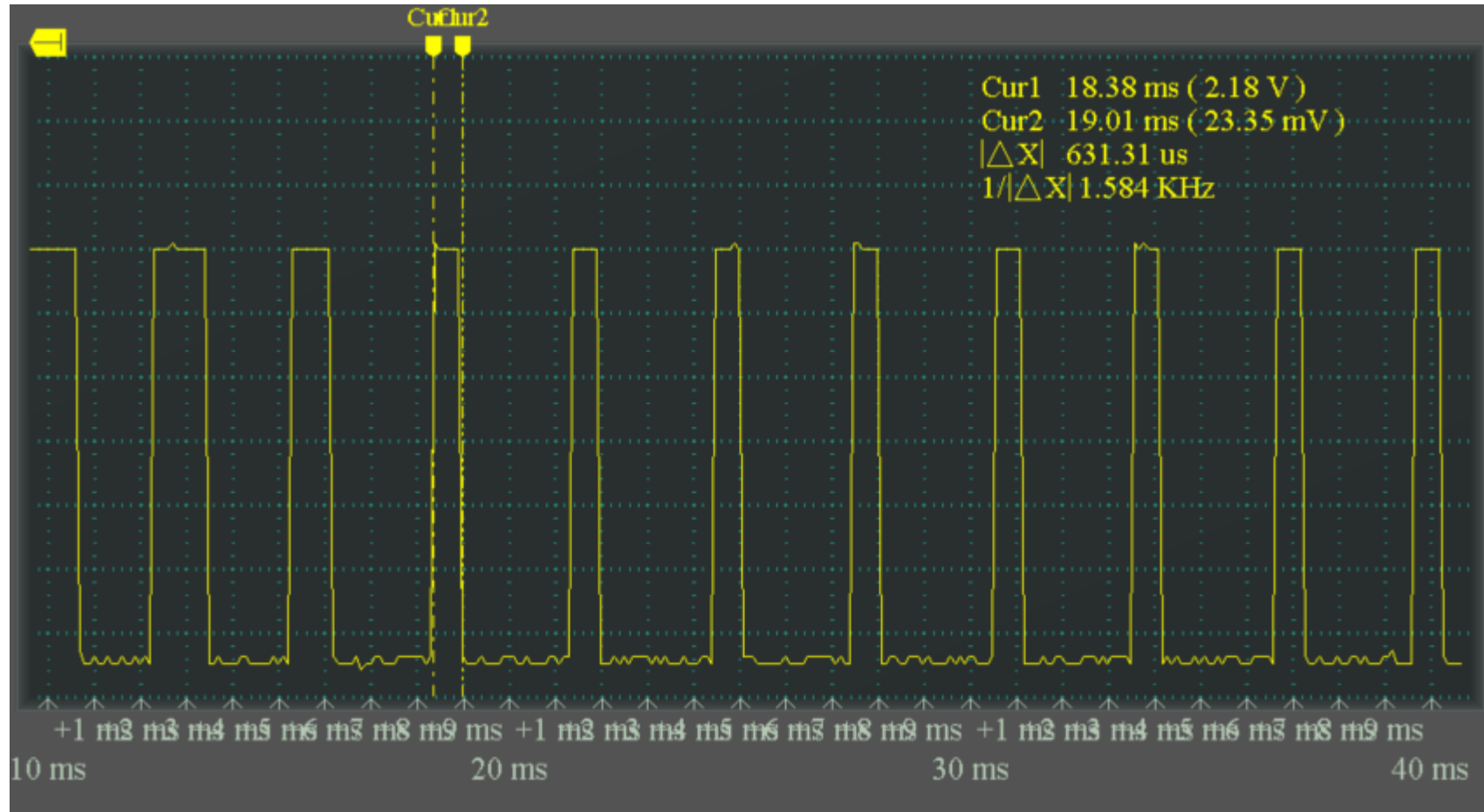
Complex Setting: Ramp_Down Stage

- Duty_Bright = 80, Ramp_down = 10, hence duty bright is reduced from 80 to 20 cycle, pass $(80-20)/10 = 6$ periods
- Start from timeline t1 = 0ms, end to timeline t2 = $6 * [100 * (1/32\text{KHz})] = 18.75 \text{ ms}$. Real is $6 * 3 = 18\text{ms}$



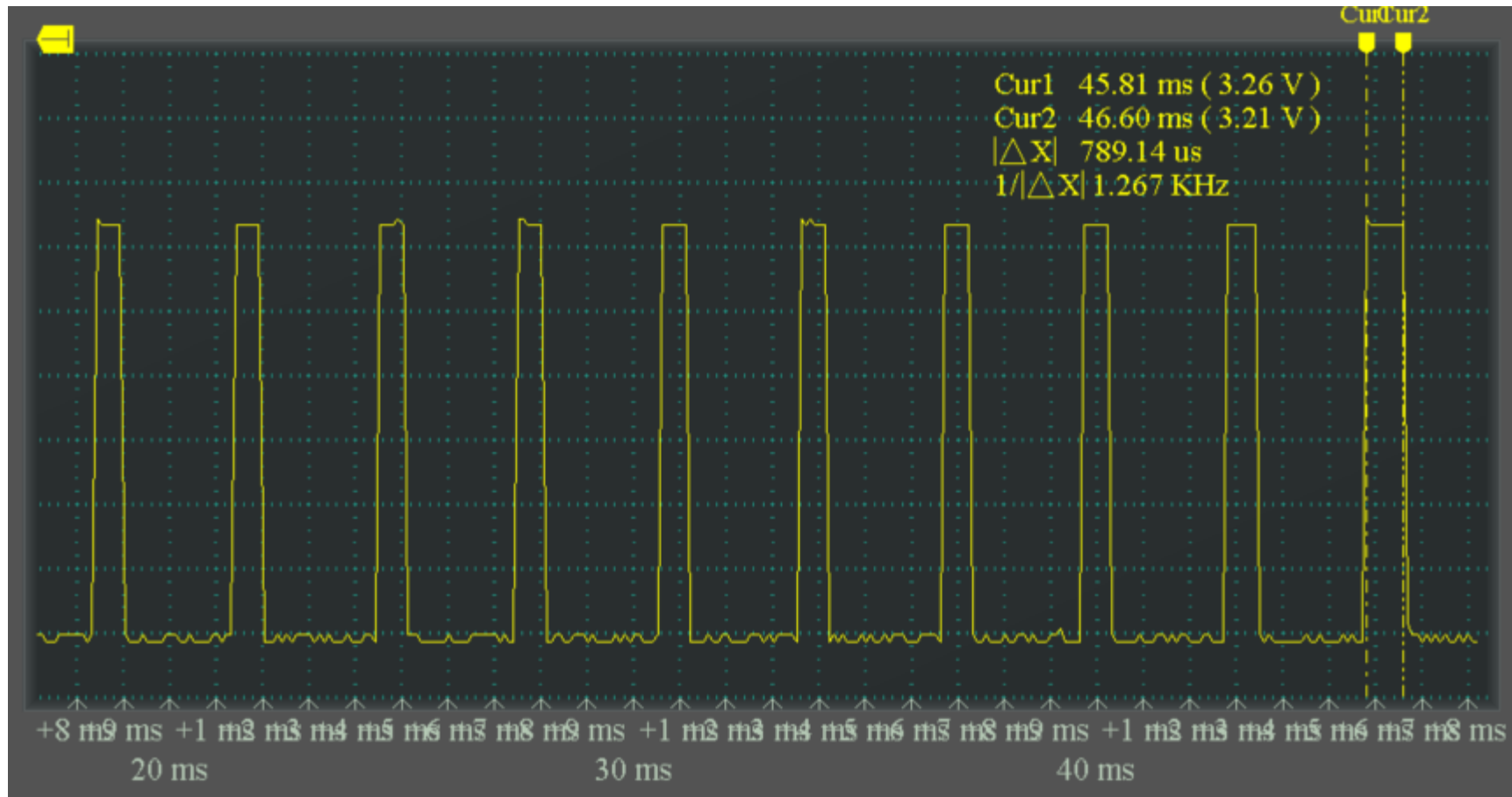
Complex Setting: Duty_Dull

- $\text{Duty_Dull} = 20 \text{ clock cycle, width equals } 20 \cdot (1/32\text{KHz}) = 625\mu\text{s}.$
- Hold Dull stage, from timeline $t_3 = 7 \cdot [100 \cdot (1/32\text{KHz})] = 21.875\text{ms}$. Keep $\text{Hold_dull} = 8 \text{ period}$, duration is 25 ms



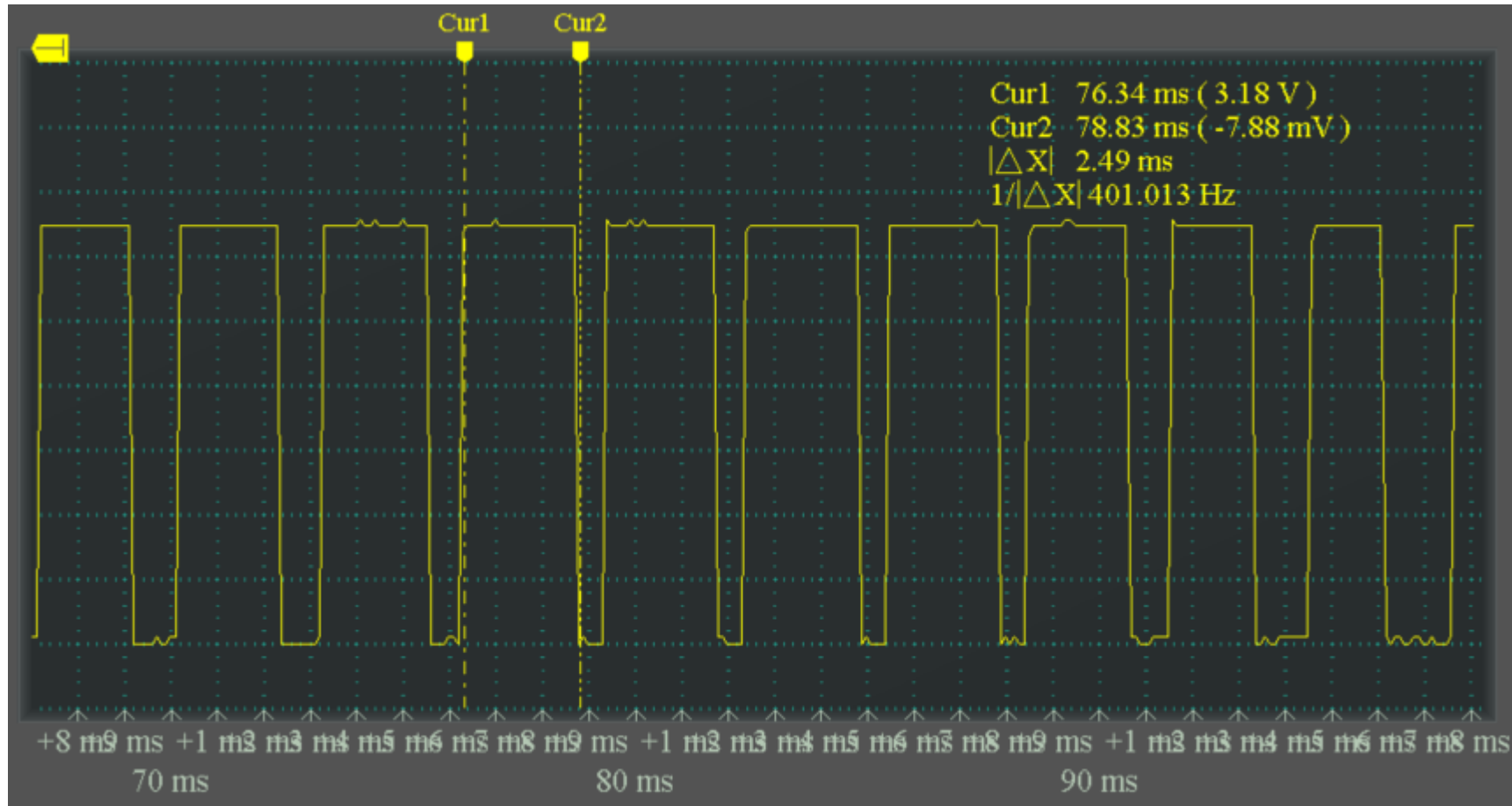
Complex Setting: Hold Dull Stage

- $\text{Hold_dull} = 8$, from 2nd, keep 8 cycles DULL state. High pulse width = $20 * \text{clock_cycle} = 20 * (1/32\text{kHz}) = 625\text{us}$
- From 9th, duty_dull is increased. timeline = $(6+8+1) * 100 * (1/32\text{kHz}) = 46.875\text{ms}$. Its width = $20+5$ (clock cycle) = $25 * (1/32\text{kHz}) = 781\text{us}$



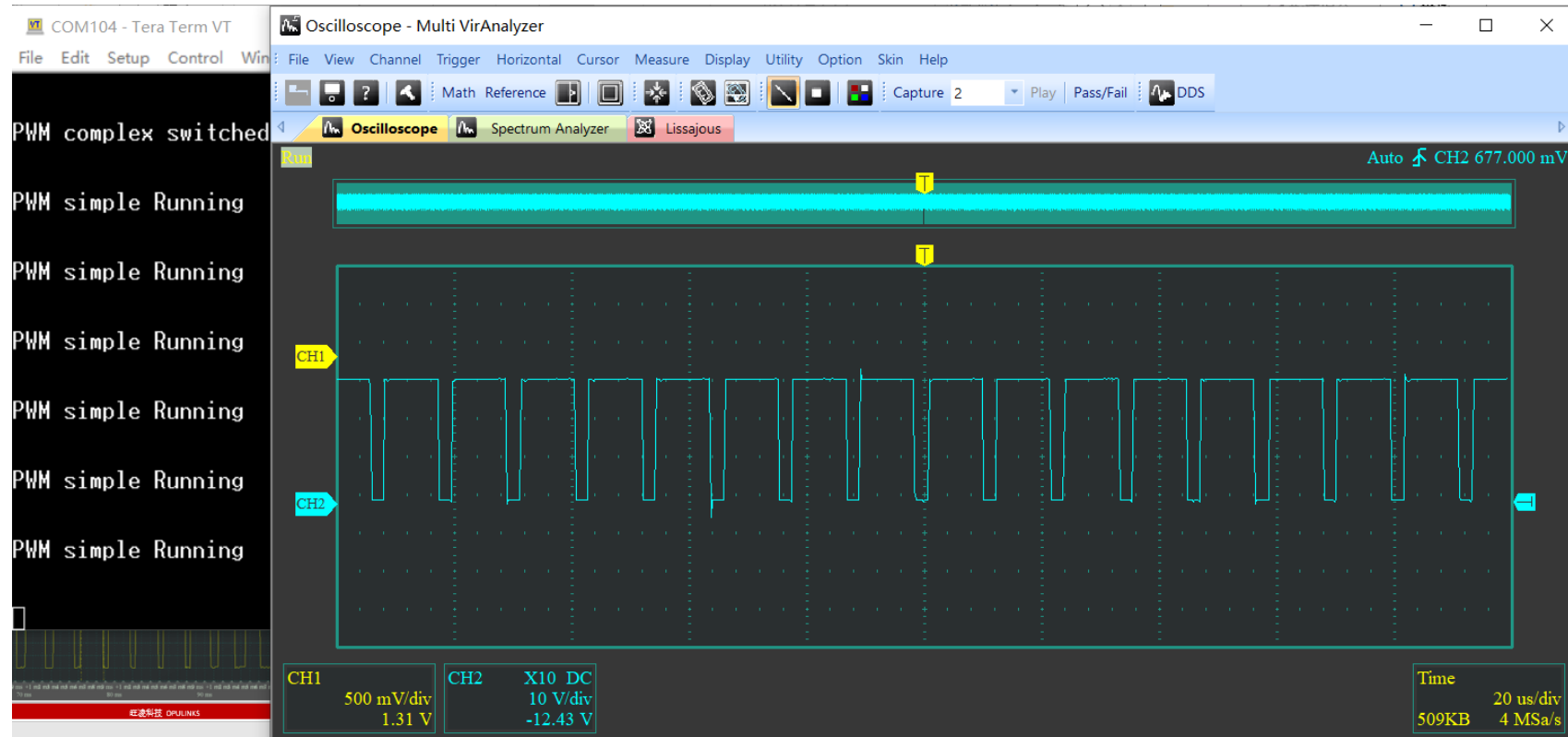
Complex Setting: Ramp up procedure

- Start time = 46.875ms. Ramp_up = 5, period count = $(80-20)/5 = 12$, end timeline = $(6+8+12) * 100 * (1/32\text{KHz}) = 81.25\text{ms}$
- “Bright” pulse duration = 80 clock cycle = 2.5ms, Hold_Bright=4, keep $4 * 100 * (1/32\text{KHz}) = 12.5\text{ ms}$ duration. Real is $4 * 3\text{ms} = 12\text{ ms}$.



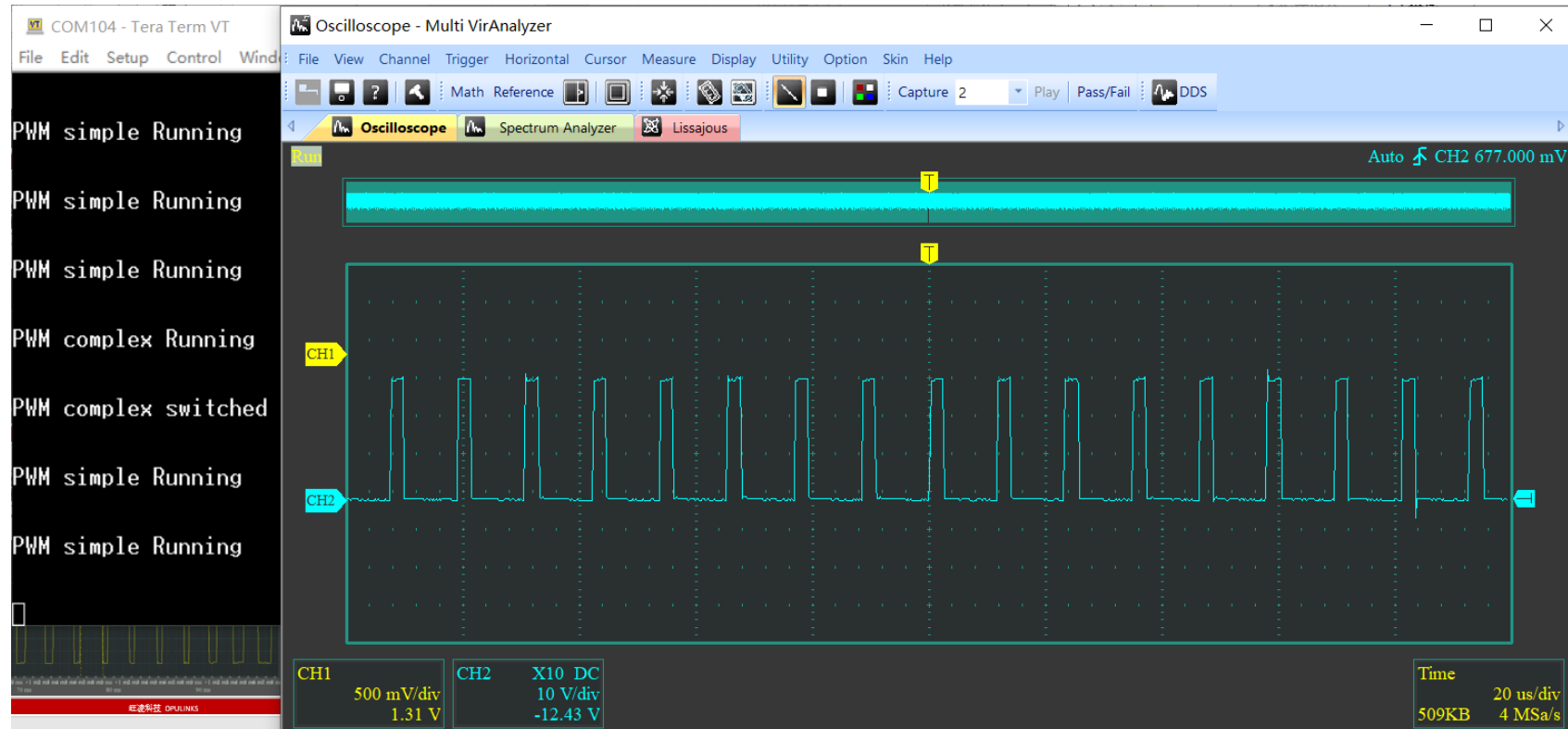
Complex Example: 80% Duty Ratio

- Period = 255, target DutyBright = 204, Hold_Bright = 1, Hold_Dull = 1



Complex Example: 20% Duty Ratio

- Period = 255, target DutyBright = 51, Hold_Bright = 1, Hold_Dull = 1





Application Notice

- In one application PWM port clock source shall be one choice.
- Both “Simple Mode “ and “Complex mode” can generate regular square wave which duty rate is fix value
- “simple mode” duty rate precise is 1%.
- “complex mode” setting can get higher precision duty rate (< 1%), depends on PWM waveform frequency.
- Roughly output waveform freq and duty precise have inverse relation. The formula is:

$(f_{\text{clock}}/f_{\text{pwm}})*p_{\text{duty}} > 1$ here f_{clock} is clock souce freq, f_{pwm} is output waveform frequency, p_{duty} is duty rate.

In one words, if f_{pwm} is higher then duty rate precise will be lower.

- Maximum output waveform frequency is half of clock source frequency.
- Minimum output waveform frequency is clock source frequency div 511.





Reference

1. Pinmux Tool use guide: [OPL1000-pinmux-tool-user-guide.pdf](#)
2. [OPL1000 HDK github site](#)
3. OPL1000 A1 datasheet: [OPL1000-DS-NonNDA.pdf](#)
4. OPL1000 A1 HDK document: [OPL1000-HDK.pdf](#)
5. A1 Devkit module board schematic: [OPL1000A1_MODULE_MONPOLE_2018_09_27.pdf](#)
6. A1 Devkit mother board schematic: [A1_MODULE_MOTHER_BOARD_V1_2_2018_10_02.pdf](#)

