Microcontroller Training

Controller: TC1796

www.infineon.com/microcontroller

MICROCONTROLLERS

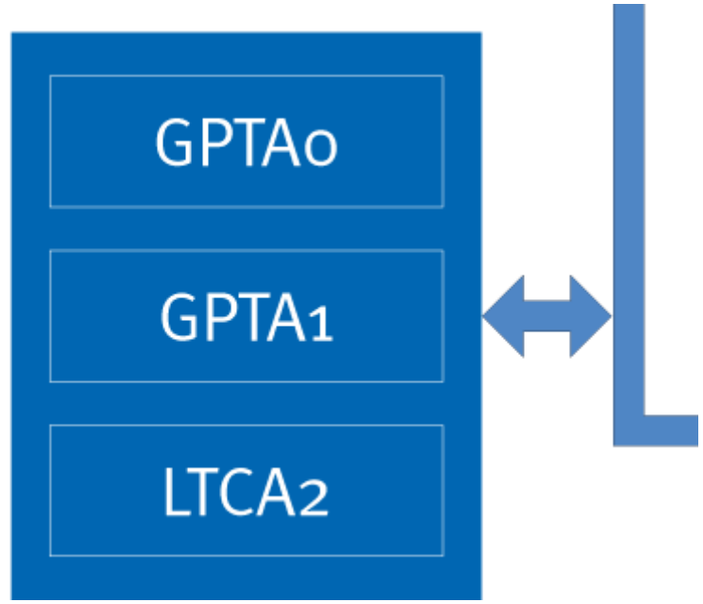
Never stop thinking

Exercise 7: PWM

PWM

In this exercise you will develop an application that generates a sinusodial complementary 3-phase PWM (Puls Width Modulation) signal with an update frequency of 20KHz using the General Purpose Timer Array (GPTA).

The GPTA provides a set of timer, compare and capture functionalities that can be flexibly combined to form signal measurement and signal generation units. They are optimized for tasks typical of engine, gearbox, and electrical motor control applications, but can also be used to generate simple and complex signal waveforms needed in other industrial applications.
The TC1796 contains two General Purpose Timer Arrays (GPTA0 and GPTA1) with identical functionality, plus an additional Local Timer Cell Array (LTCA2).



Exercise 7: PWM

GPTA Features

Each of the General Purpose Timer Arrays (GPTA0 and GPTA1) provides a set of hardware modules required for high-speed digital signal processing:

Clock Generation Unit

- Filter and Prescaler Cells (FPC) support input noise filtering and prescaler operation.
- Phase Discrimination Logic units (PDL) decode the direction information output by a rotation tracking system.
- Duty Cycle Measurement Cells (DCM) provide pulse-width measurement capabilities.
- A Digital Phase Locked Loop unit (PLL) generates a programmable number of GPTA module clock ticks during an input signal's period.

Signal Generation Unit

- Global Timer units (GT) driven by various clock sources are implemented to operate as a time base for the associated Global Timer Cells.
- Global Timer Cells (GTC) can be programmed to capture the contents of a Global Timer on an external or internal event. A GTC may also be used to control an external port pin depending on the result of an internal compare operation. GTCs can be logically concatenated to provide a common external port pin with a complex signal waveform.
- Local Timer Cells (LTC) operating in Timer, Capture, or Compare Mode may also be logically tied together to drive a common external port pin with a complex signal waveform. LTCs, enabled in Timer Mode or Capture Mode, can be clocked or triggered by various external or internal events.

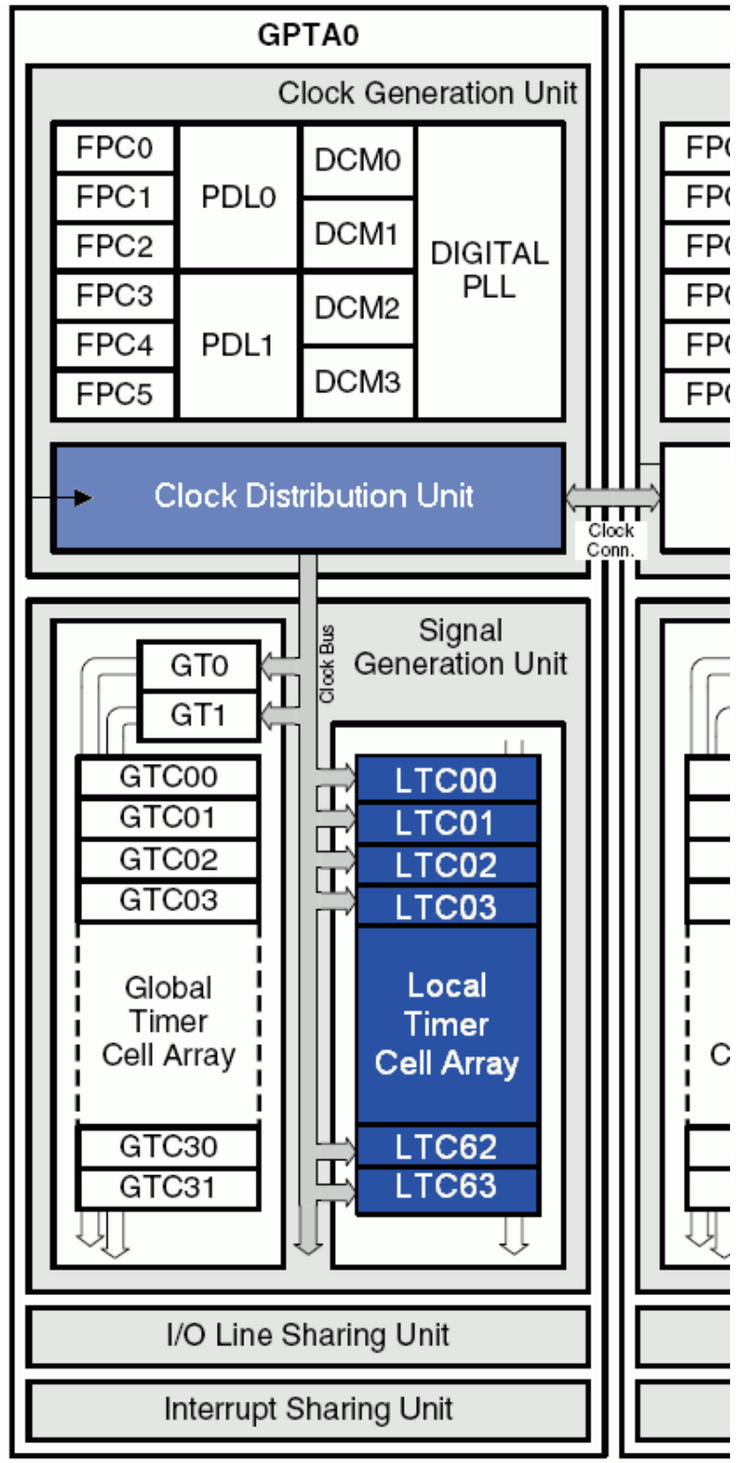
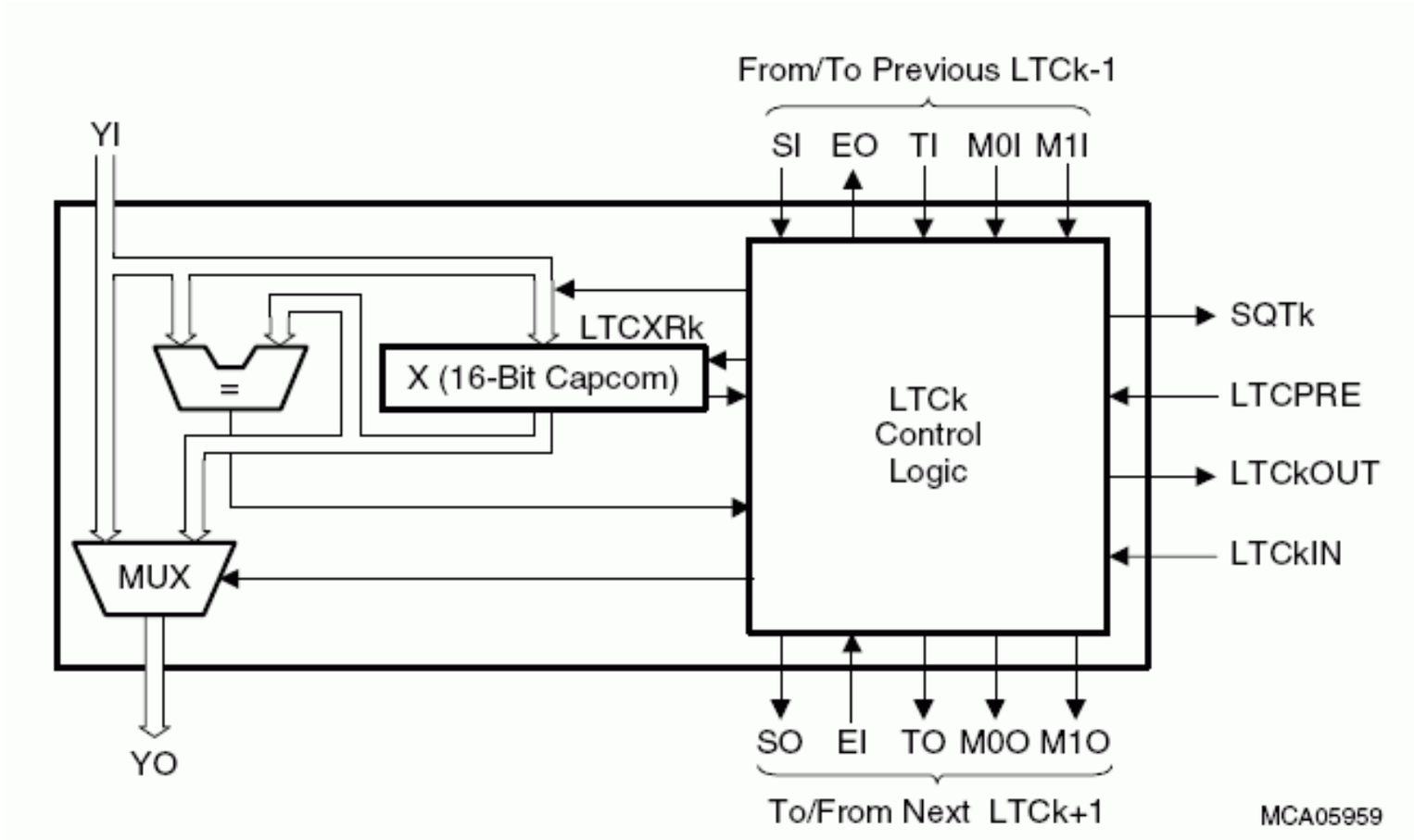


Fig.1 GPTA Block Diagram

Exercise 7: PWM

LTC Functionality

- Local Timer Cell (LTC)
- 64 independent units
- Three basic operating modes (Timer, Capture and Compare) for 63 units
- Special compare modes for one unit
- 16-bit data width
- f_{GPTA} maximum resolution
- $f_{GPTA}/2$ maximum input signal frequency



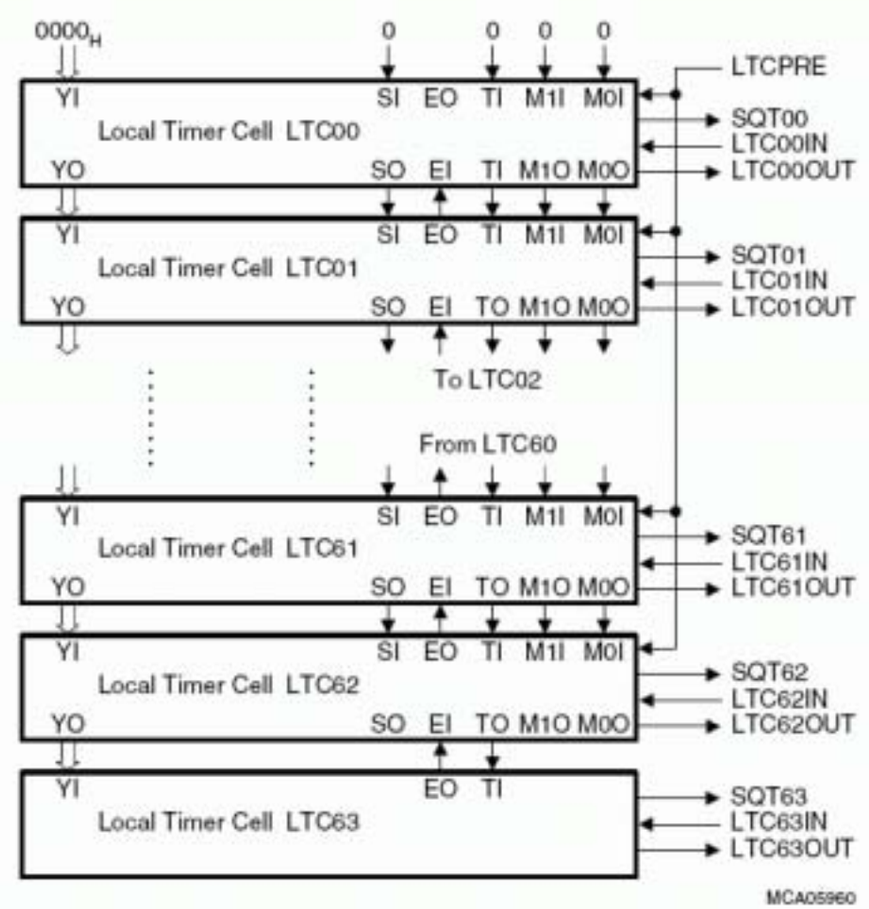


Fig.2 Architecture of Local Timer Cells and Interconnections between LTCs

Exercise 7: PWM

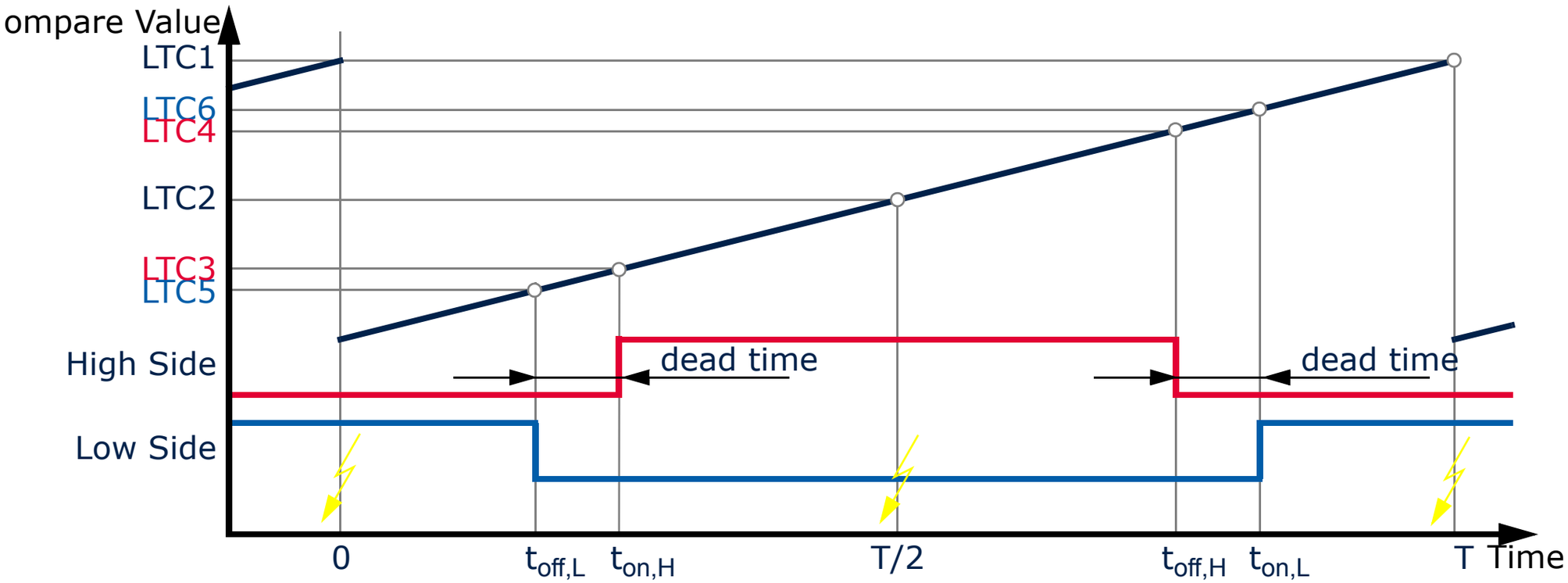


Fig.3 PWM Generation with Local Timer Cells

	Value	Cell	Output Control Mode	Port
Reset Timer	-	LTC0	-	-
Compare Period	T	LTC1	Interrupt: adjust $t_{off,H}$, $t_{on,L}$	-
Compare Mid Period	T/2	LTC2	Interrupt: adjust $t_{on,H}$, $t_{off,L}$	-
Compare High side On	$t_{on,H}$	LTC3	Set output by a local event	-
Compare High side Off	$t_{off,H}$	LTC4	Reset output by a local event or copy the previous cell action	P2.8
Compare Low Side Off	$t_{on,L}$	LTC5	Reset output by a local event	-
Compare Low Side On	$t_{on,L}$	LTC6	Set output by a local event or copy the previous cell action	P2.9

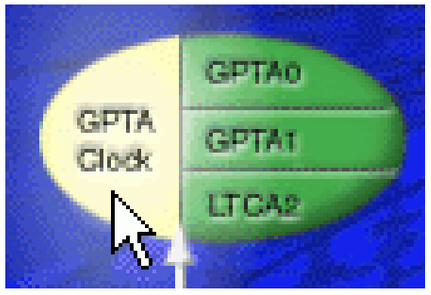
Exercise 7: PWM

1. Create a DAvE project

Open the *Windows Explorer* and create a new directory `c:\infineon\pwm`. Copy the `isr.dav` file from the previous exercise to the new directory and rename the file to `pwm.dav`.

2. Open the GPTA clock properties

Click on **GPTA Clock** in the project window to open the GPTA Clock properties.

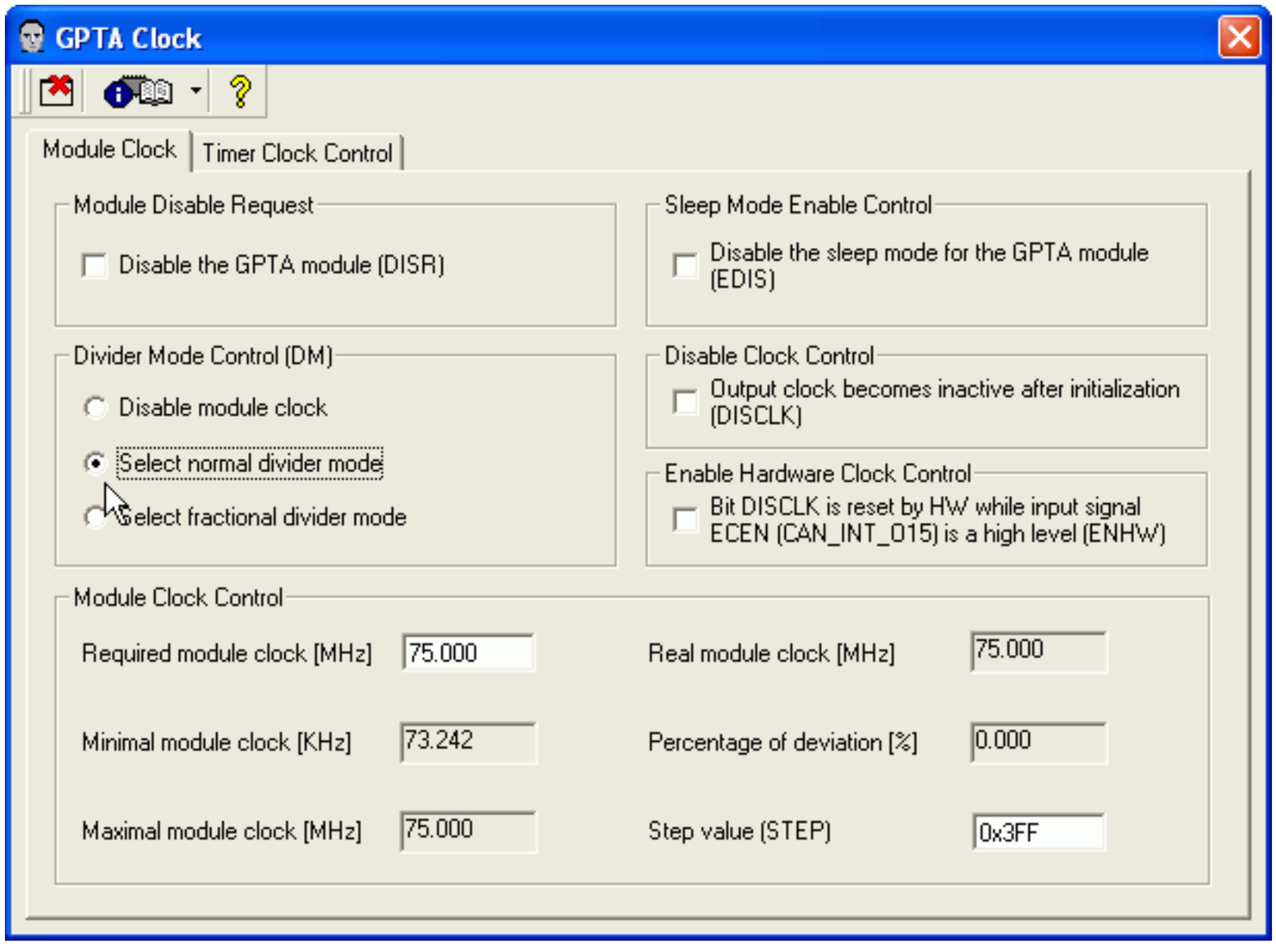


Exercise 7: PWM

3. Set up the GPTA clock properties

On the **Module Clock** page

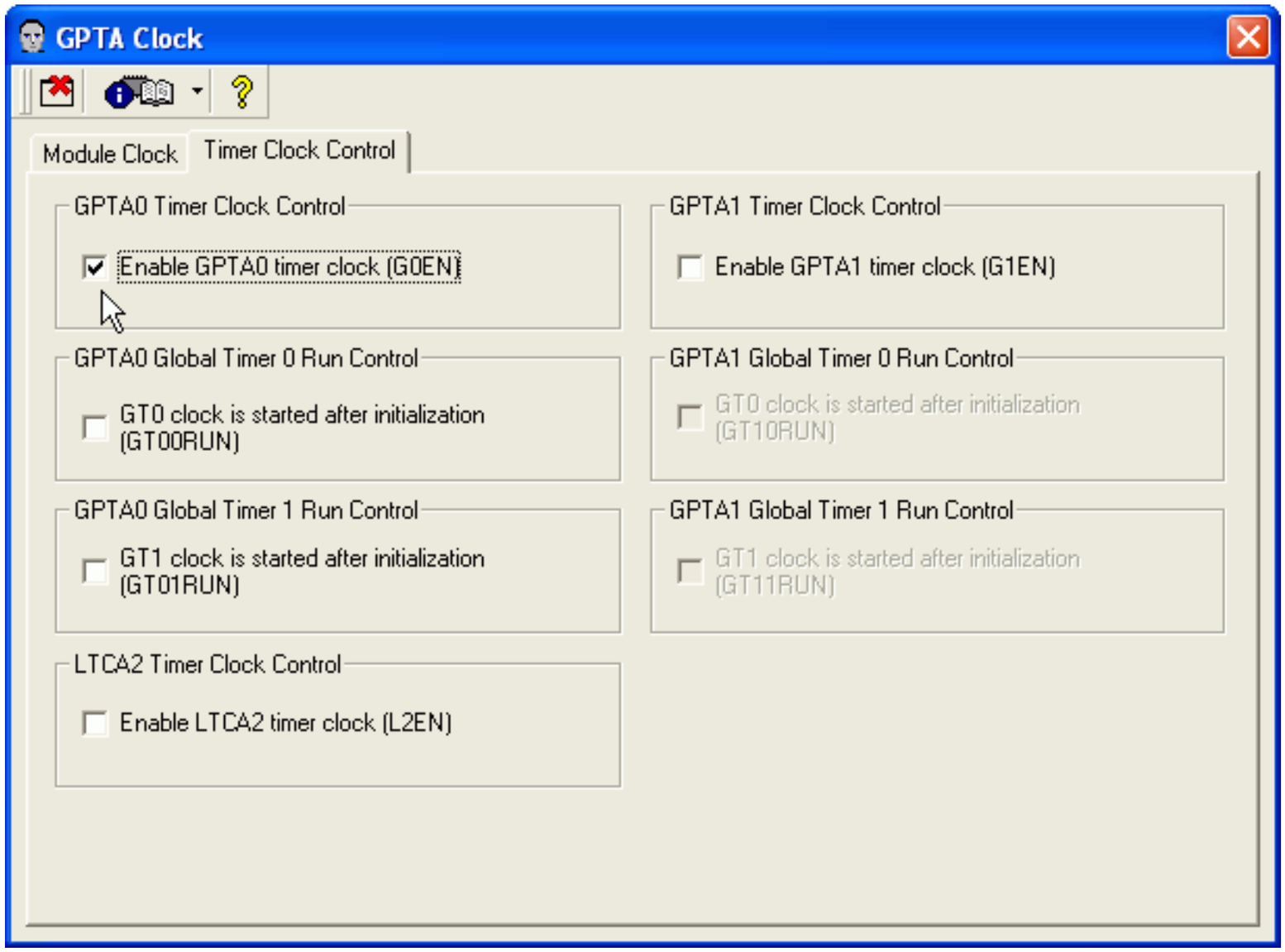
- Select **Select normal divider mode**.
- Enter **Required module clock [MHz]** to f_{GPTA} = 75 MHz.



Exercise 7: PWM

On the **Timer Clock Control**

■ Check **Enable GPTA0 timer clock (G0EN)**.

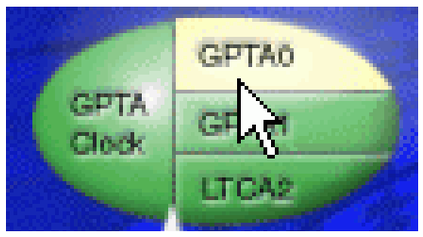


The GPTA clock is completely configured. Click the **Close** icon on the dialog toolbar to close the **GPTA Clock** dialog.

Exercise 7: PWM

4. Open the GPTA0 properties

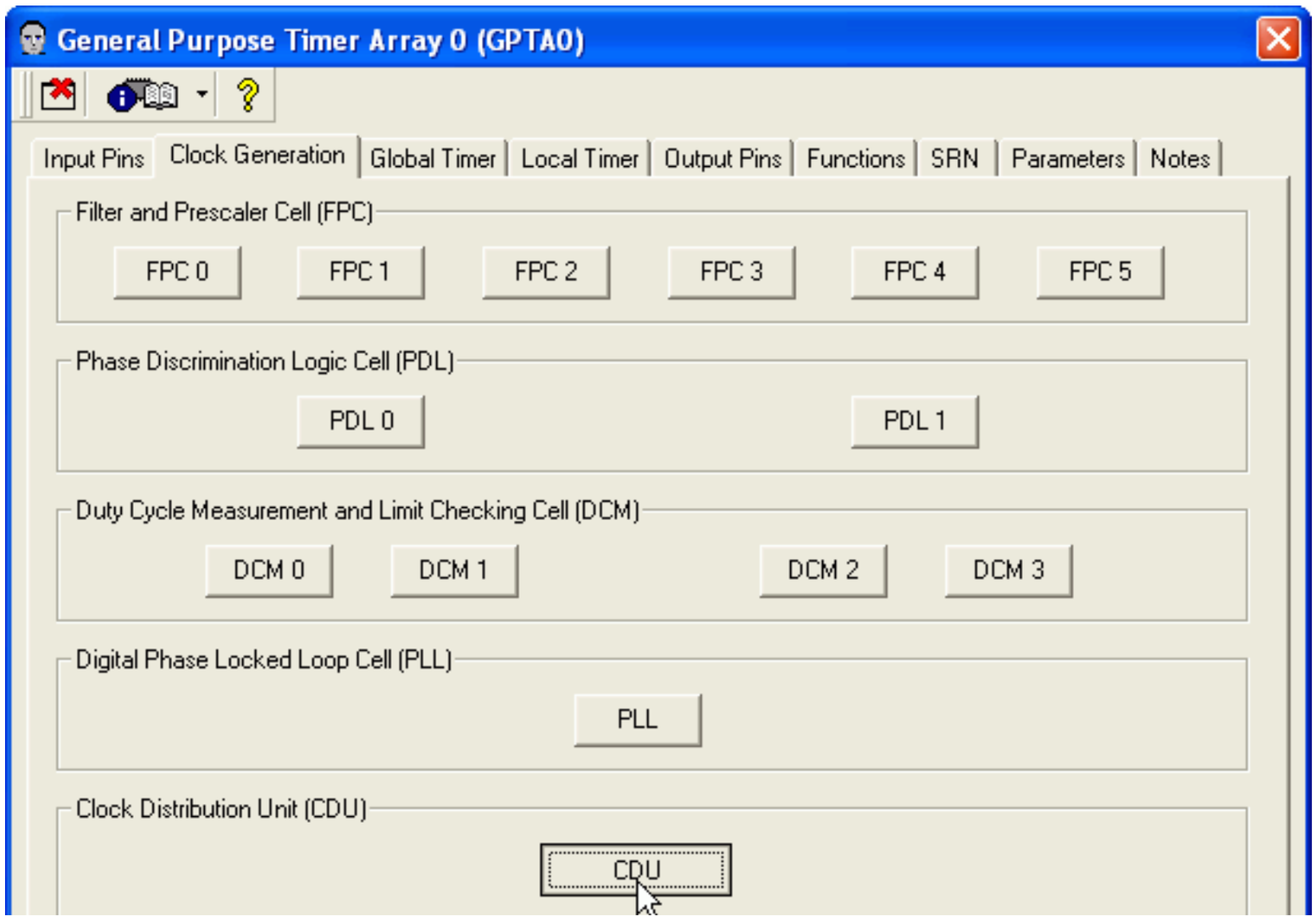
Click on **GPTA0** in the project window to open the GPTA0 properties.



Exercise 7: PWM

5. Set up the GPTA0 properties

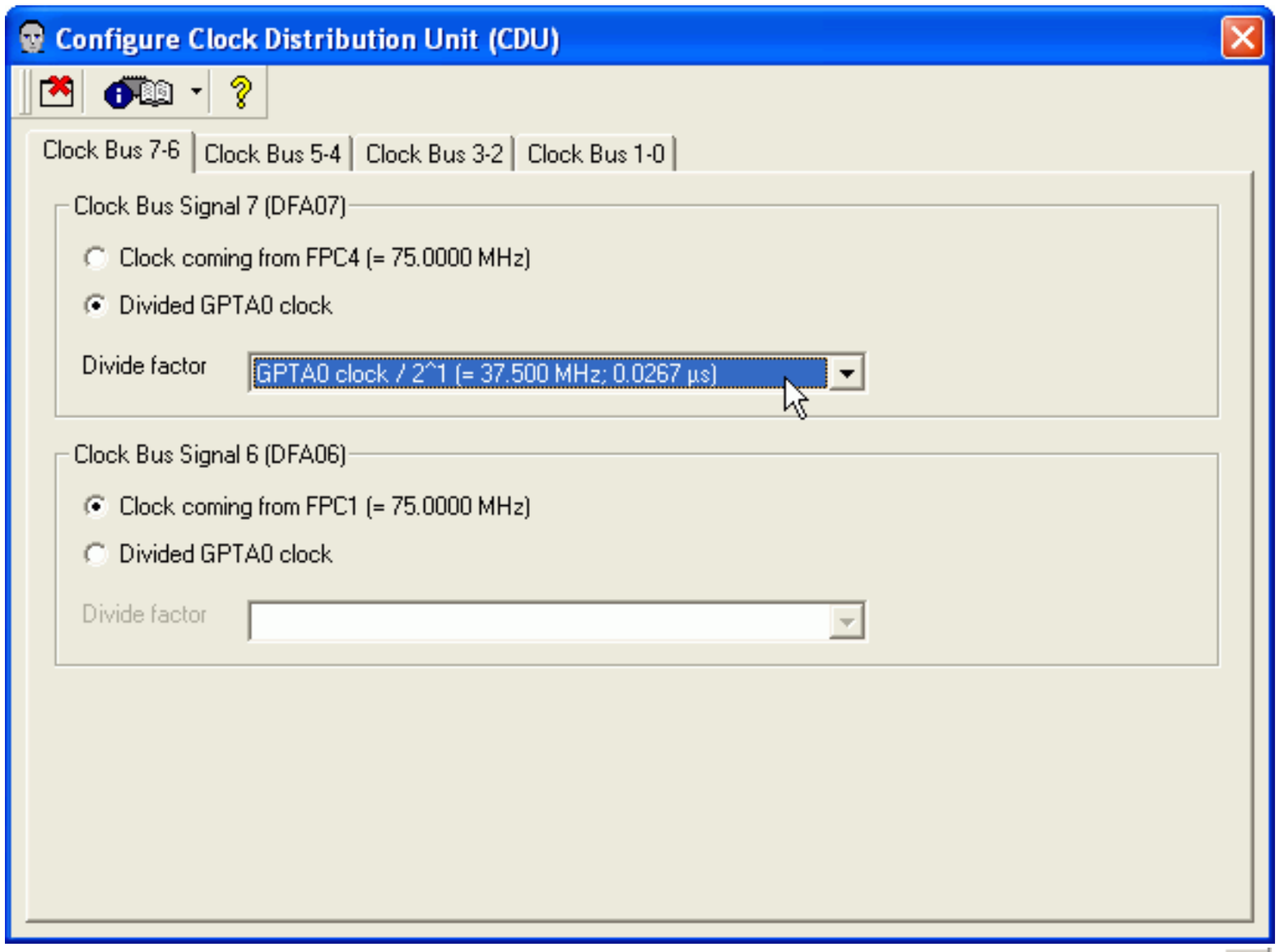
On the **Clock Generation** page click **CDU** to open the Clock Distribution Unit properties.



Exercise 7: PWM

On the **Clock Bus 7-6** page

- At **Clock Bus Signal 7 (DFA07)** select **Divided GPTA0 clock** and choose the Divide factor **GPTA0 clock / 2^1**. The maximum input frequency of the local timer cells is $f_{GPTA}/2$.

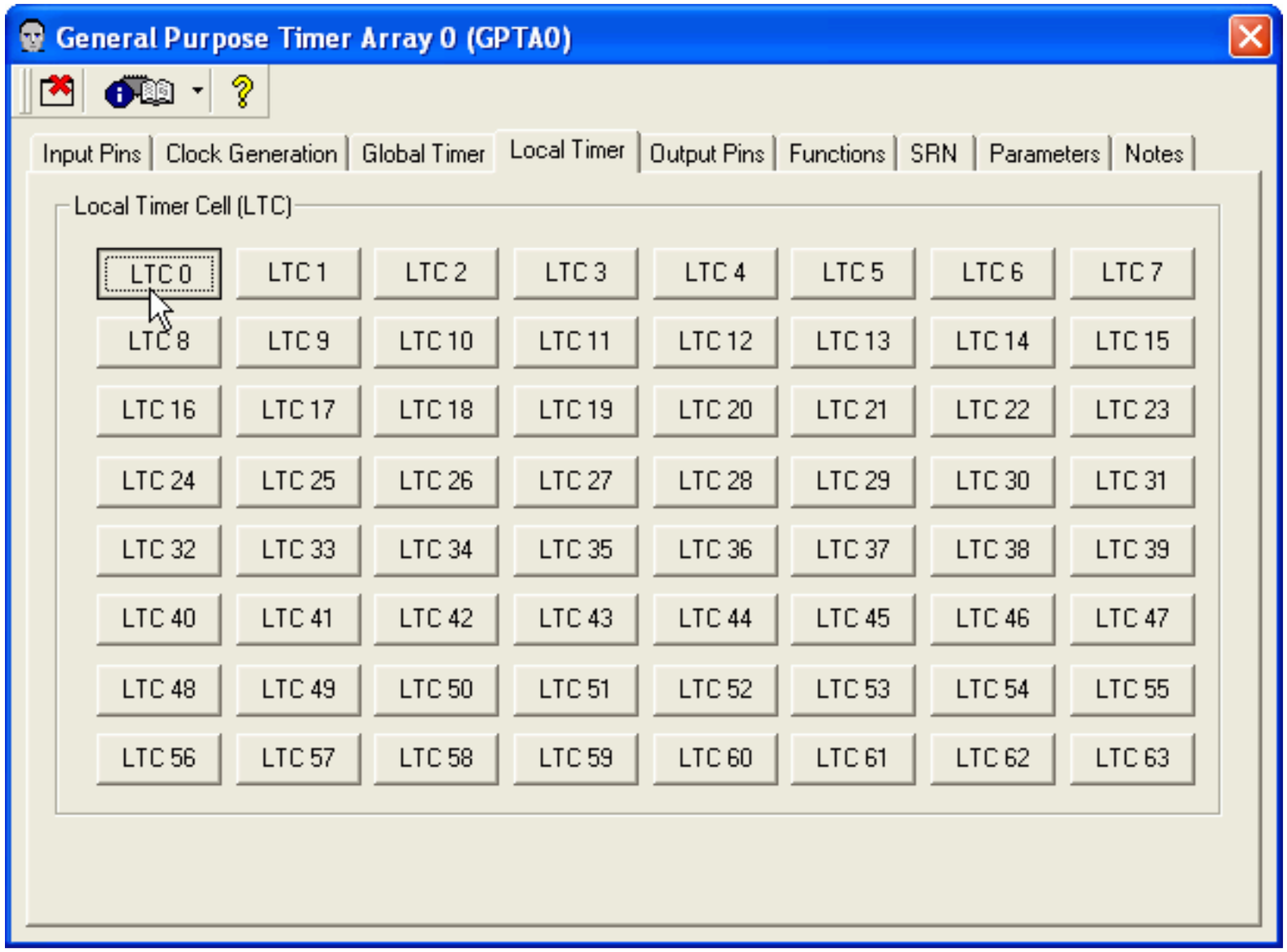


The Clock Distribution Unit is completely configured. Click the **Close** icon on the dialog toolbar to close the **Configure Clock Distribution Unit (CDU)** dialog.

Exercise 7: PWM

6. Set up the GPTA0 properties. Local Timer

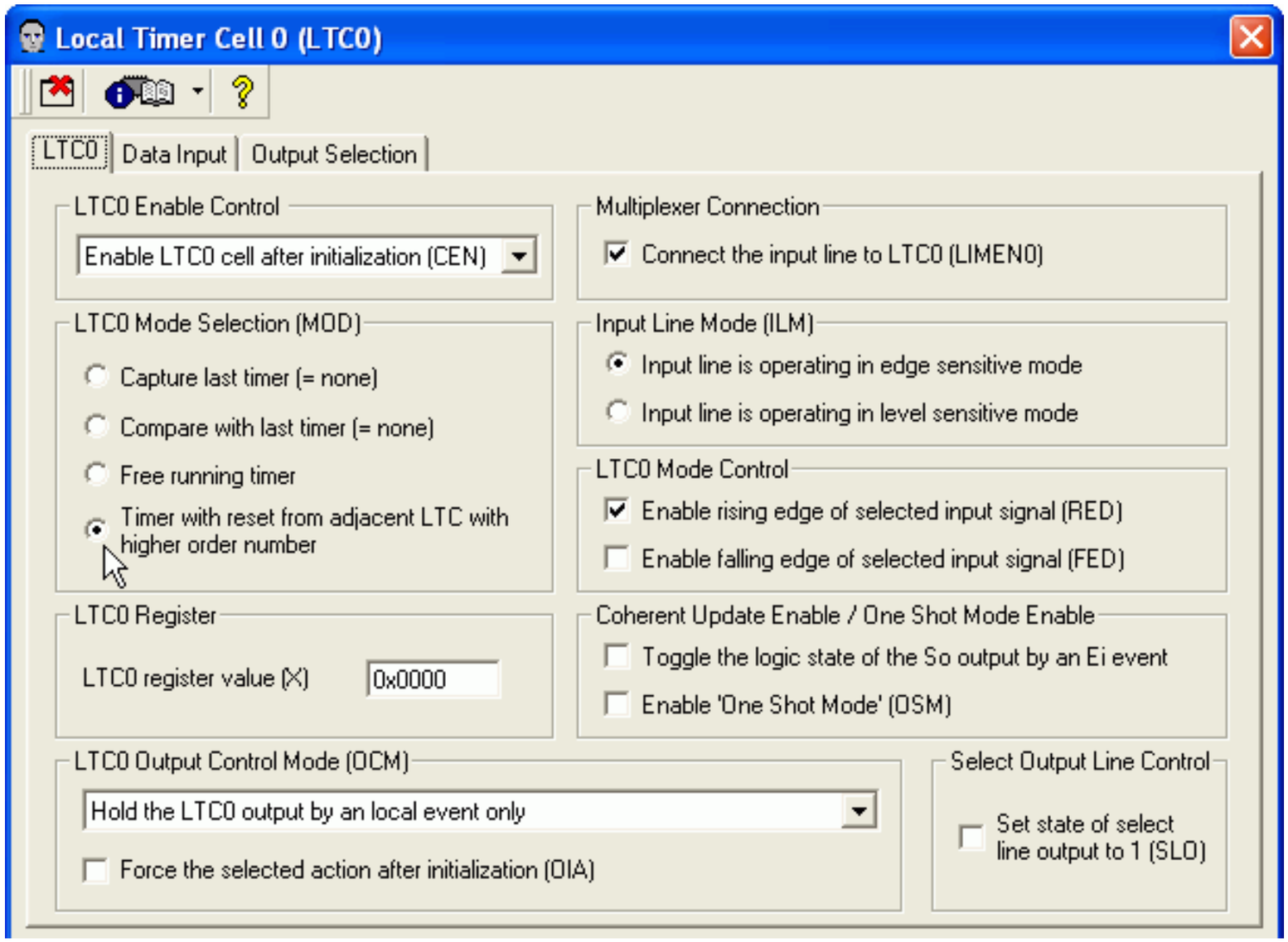
On the **Local Timer** page click **LTC0** to open the Local Timer Cell 0 properties.



Exercise 7: PWM

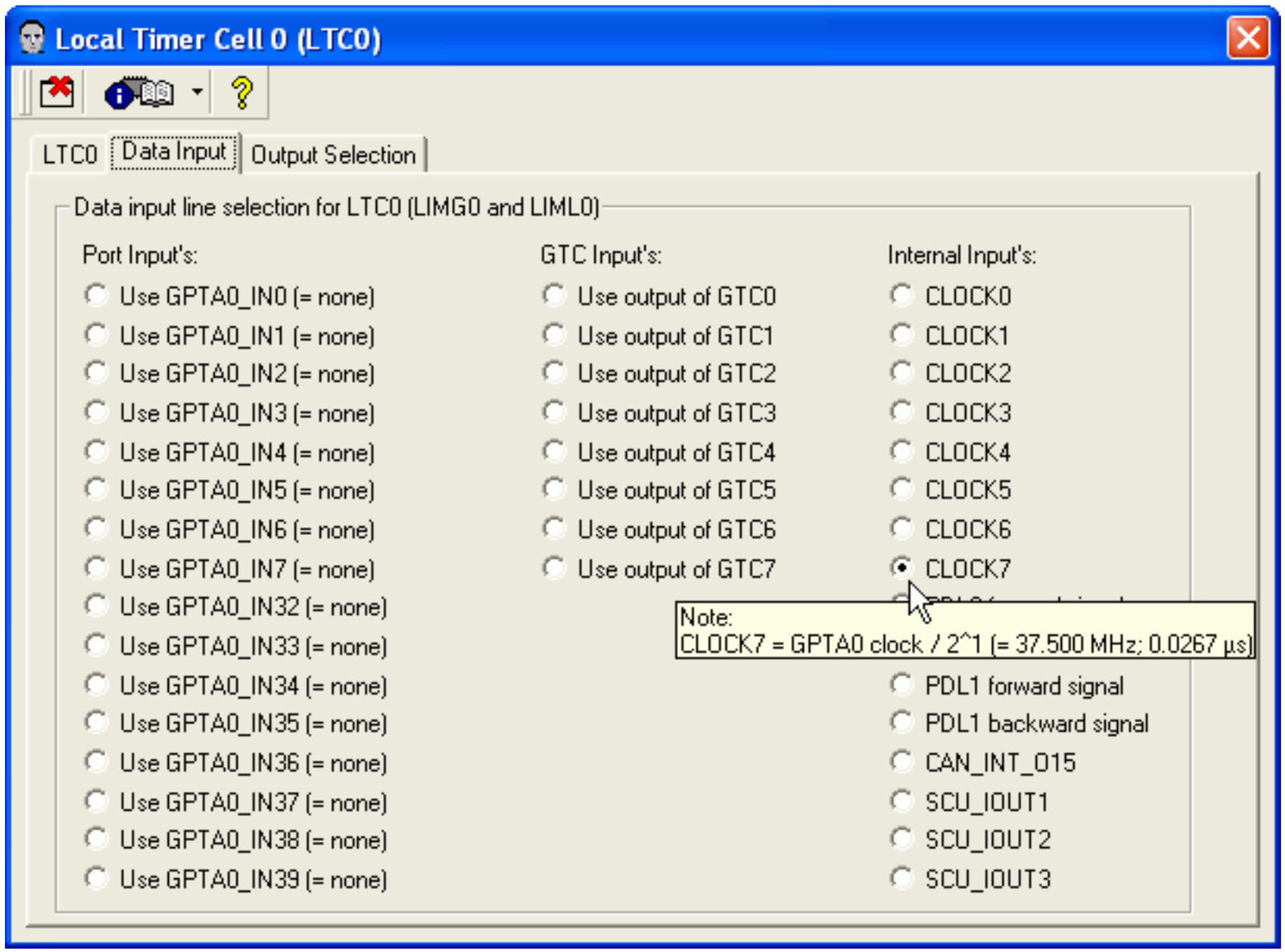
On the **LTC0** page configure the cell as reset timer:

- Choose **Enable LTC0 cell after initialization (CEN)**,
- Select **Timer with reset from adjacent LTC with higher order number**. When the timer value in LTC0 matches the compare value stored in LTC1, the Event Out signal from LTC1 is activated and passed upstream to LTC0 as the Event In signal which then resets the timer to the value of $FFFF_h$.
- Check **Connect the input line to LTC0 (LIMEN0)**. The cell will be connected to the clock bus line 7 which we configured in step 5.
- Check **Enable rising edge of selected input signal (RED)**.



Exercise 7: PWM

On the **Data Input** page of LTC0 and select the internal input **Clock7** as data input line for LTC0.

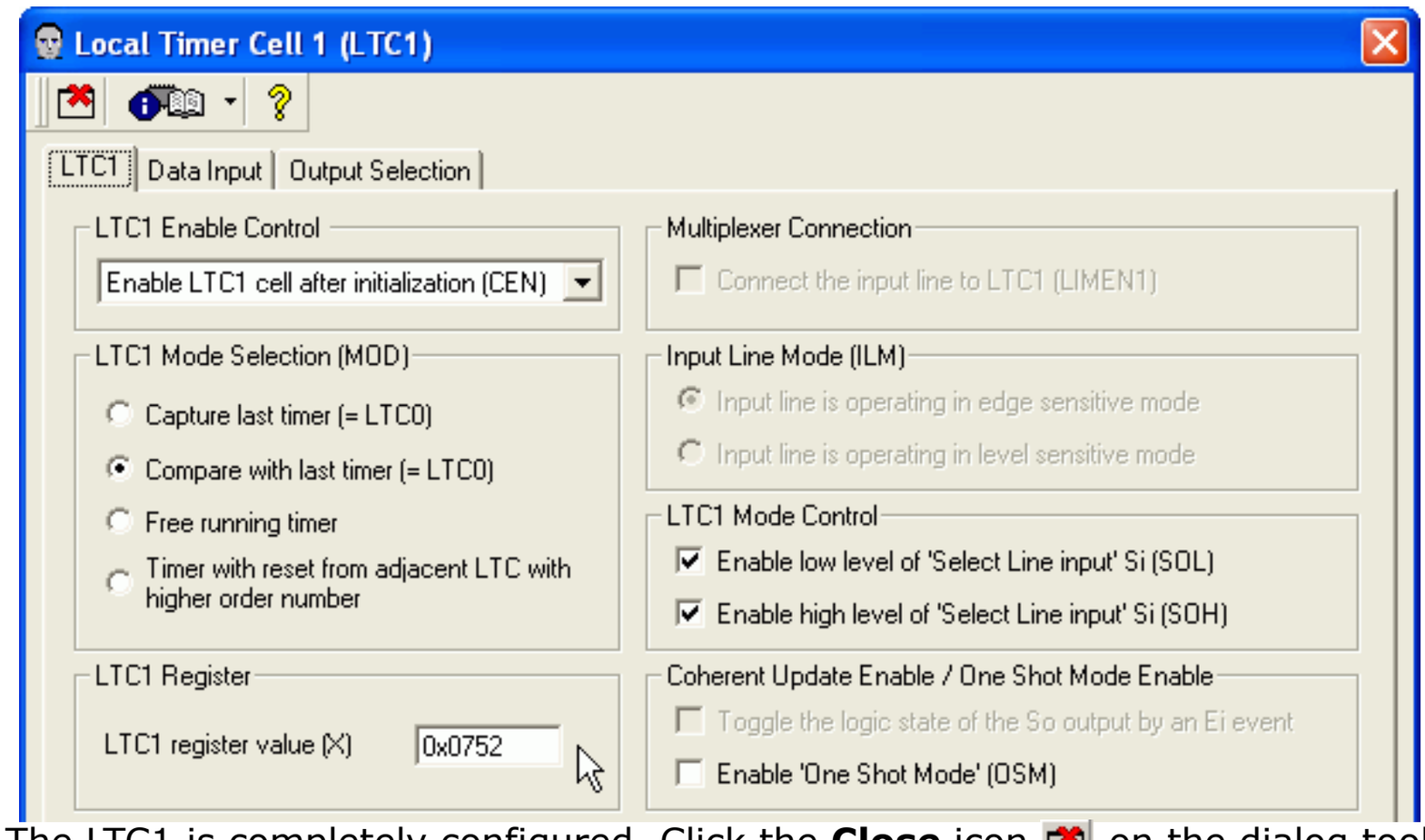


The LTC0 is completely configured as reset timer running at $f_{GPTA}/2$ = 37.5MHz. Click the **Close** icon on the dialog tool-bar to close the **Local Timer Cell 0 (LTC0)** dialog.

Exercise 7: PWM

Click **LTC1** on the **Local Timer** page. On the **LTC1** page configure the cell to compare mode:

- Choose **Enable LTC1 cell after initialization (CEN)**,
- Select **Compare with last timer (=LTC0)**,
- Check **Enable low level of 'Select Line input' Si (SOL)** and **Enable high level of 'Select Line input' Si (SOH)** to enable Compare mode in all cases,
- Set the **LTC1 register value (X)** to the period value. Since the timer LTC0 resets to -1 ($FFFF_h$) and then counts through 0 up to (and including) the compare value, the compare value used to generate a periodic interval is somewhat different than what might be expected. To generate a periodic interval of P clocks, the compare value must be set to (P-2). To get a symmetric PWM P must be even. P = 37.5MHz/20KHz = 1875. The period P will be choosen as the next higher even value 1876. The register is set to 1874 = 752_h. Enter **0x0752** in the text field and type **Return**.

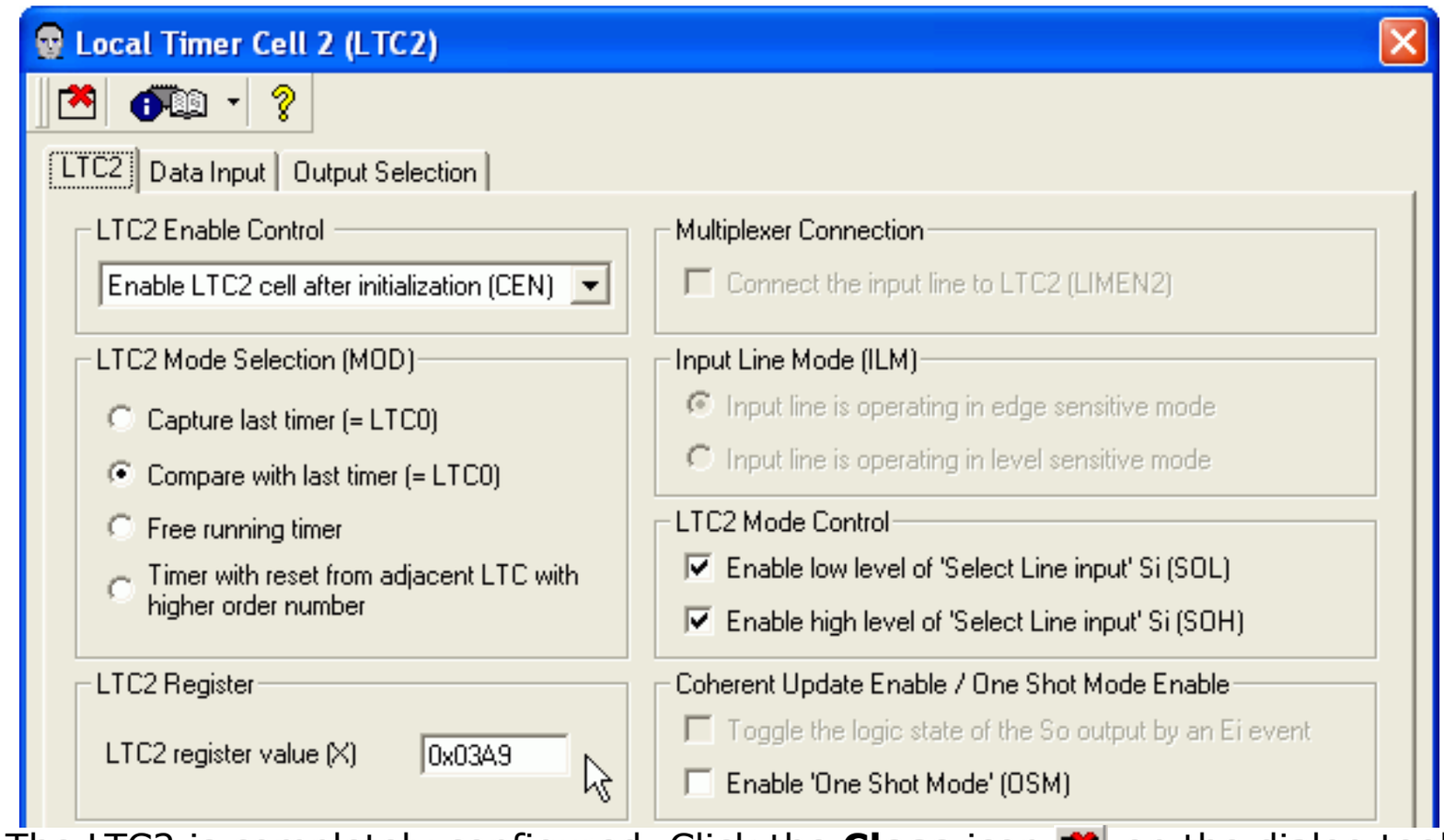


The LTC1 is completely configured. Click the **Close** icon on the dialog toolbar to close the **LTC1** dialog.

Exercise 7: PWM

Click **LTC2** on the **Local Timer** page. On the **LTC2** page and configure the cell to compare mode:

- Choose **Enable LTC2 cell after initialization (CEN)**,
- Select **Compare with last timer (=LTC0)**,
- Check **Enable low level of 'Select Line input' Si (SOL)** and **Enable high level of 'Select Line input' Si (SOH)** to enable Compare mode in all cases,
- Set the **LTC2 register value** to the mid period value -1+1876/2 = 937 = $3A9_h$. Enter **0x03A9** in the text field and type **Return**.

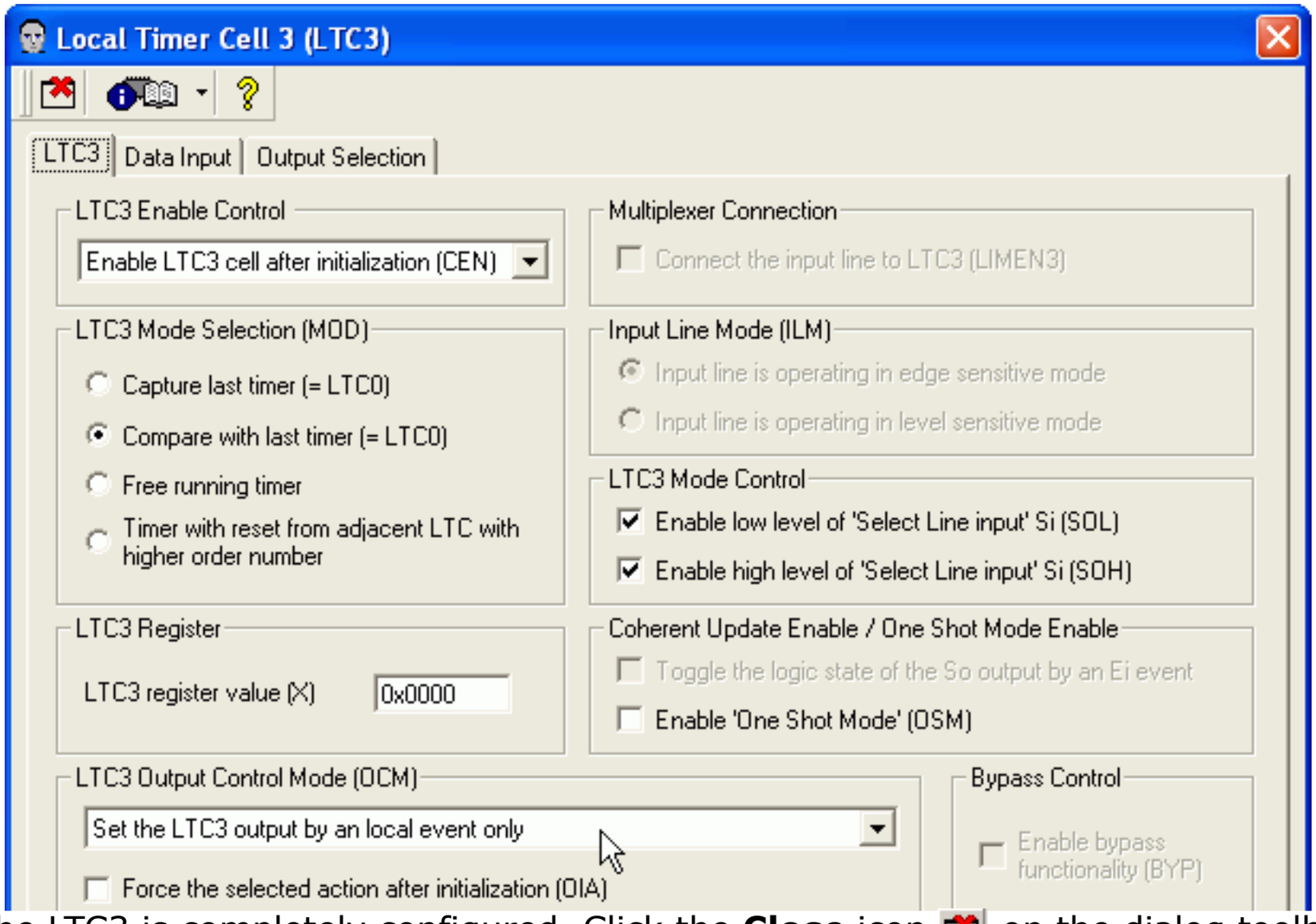


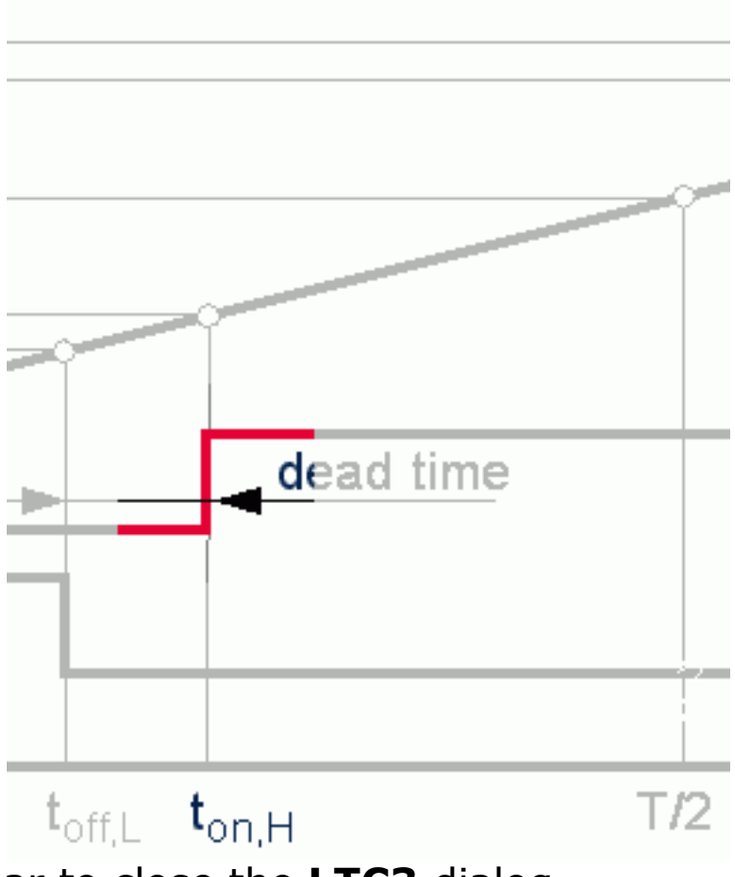
The LTC2 is completely configured. Click the **Close** icon on the dialog toolbar to close the **LTC2** dialog.

Exercise 7: PWM

To configure the phase U using LTC3/LTC4 and LTC5/LTC6 click **LTC3** on the **Local Timer** page.
On the **LTC3** page configure the cell to compare mode:

- Choose **Enable LTC3 cell after initialization (CEN)**,
- Select **Compare with last timer (=LTC0)**,
- Check **Enable low level of 'Select Line input' Si (SOL)** and **Enable high level of 'Select Line input' Si (SOH)** to enable Compare mode in all cases,
- Choose **Set the LTC3 output by an local event only** as the **Output Control Mode** to set the output to high on a compare match. The **LTC3 register value** will be calculated in the mid period interrupt routine.



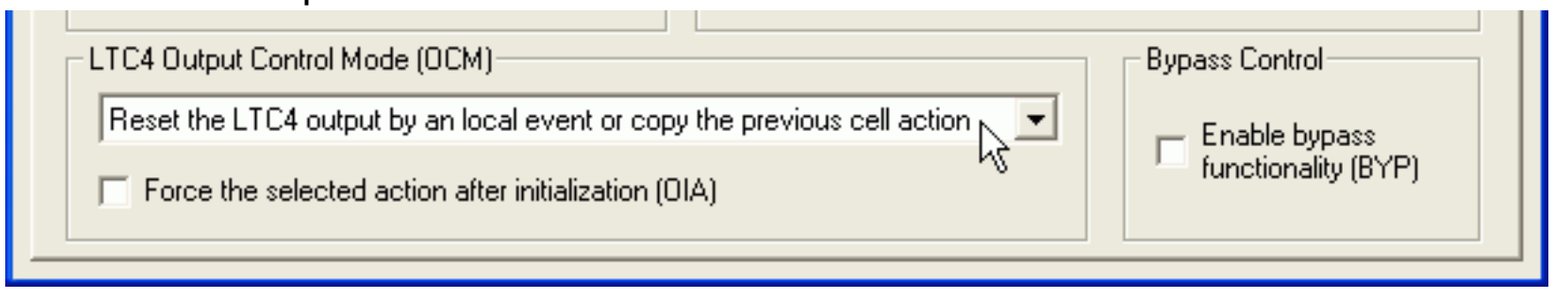


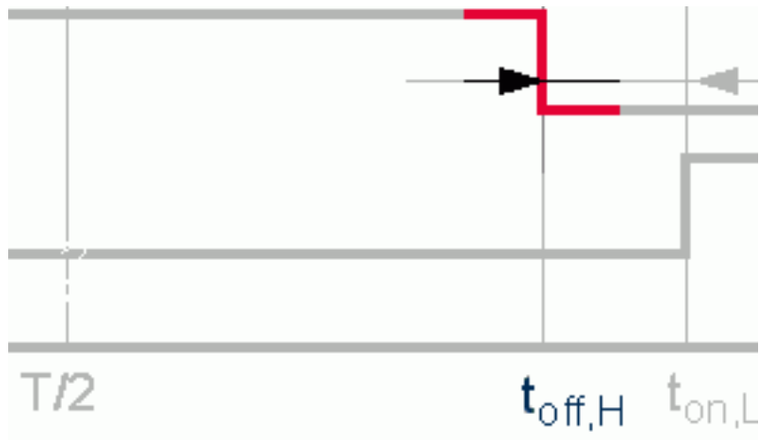
The LTC3 is completely configured. Click the **Close** icon on the dialog toolbar to close the **LTC3** dialog.

Exercise 7: PWM

Open the LTC4 to LTC6 and configure them also in compare mode

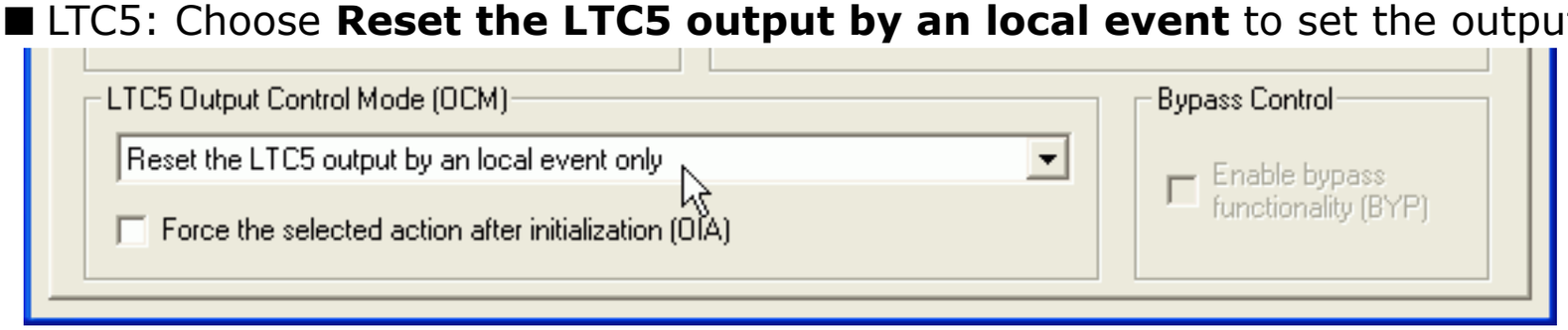
- LTC4: Choose **Reset the LTC4 output by an local event or copy the previous cell action** to set the output to passive on a compare match.

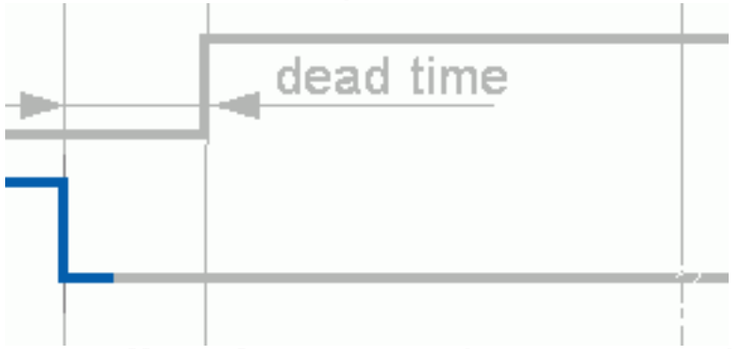




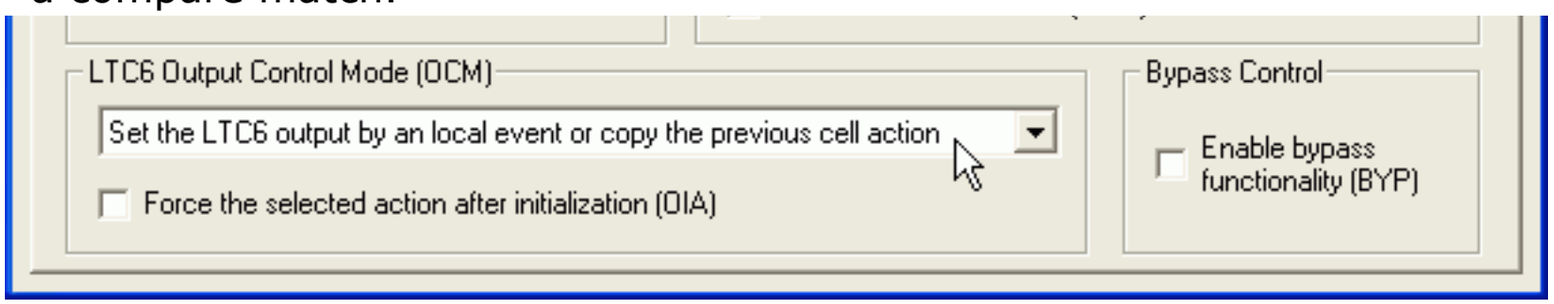
To configure the low side of phase U using LTC5/LTC6 configure the **Output Control Mode**:

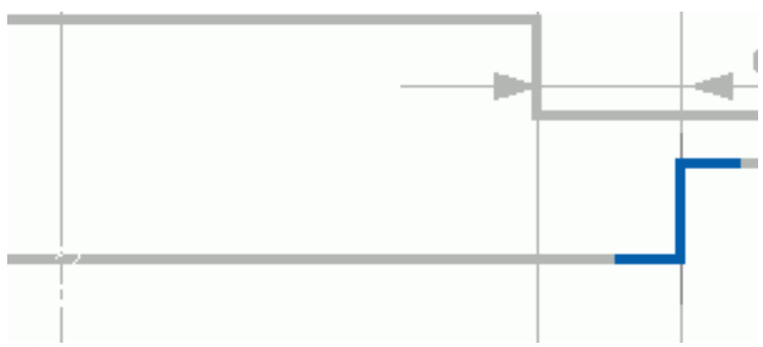
- LTC5: Choose **Reset the LTC5 output by an local event** to set the output to low on a compare match,





- LTC6: Choose **Set the LTC6 output by an local event or copy the previous cell action** to set the output to high on a compare match.



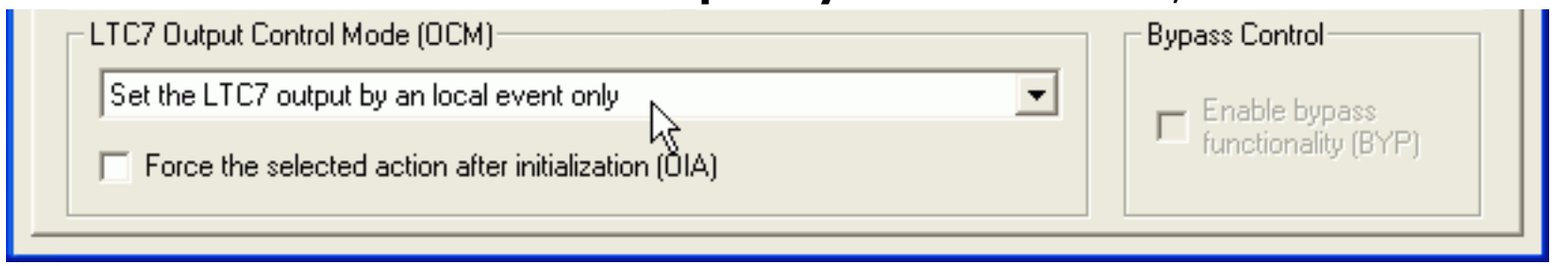


The high side and the low side of phase U are configured. Close the **LTC3** to **LTC6** dialogs.

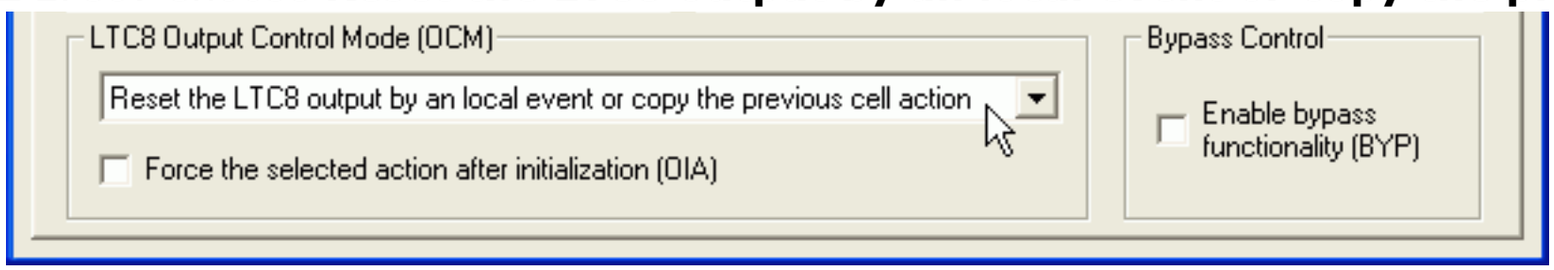
Exercise 7: PWM

For phase V open cell LTC7 to LTC10 and configure them also in compare mode:

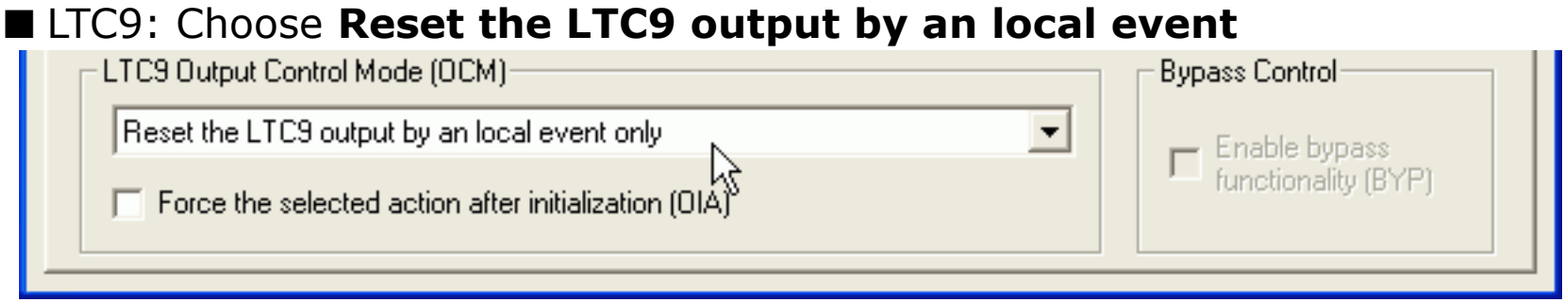
- LTC7: Choose **Set the LTC7 output by an local event**,



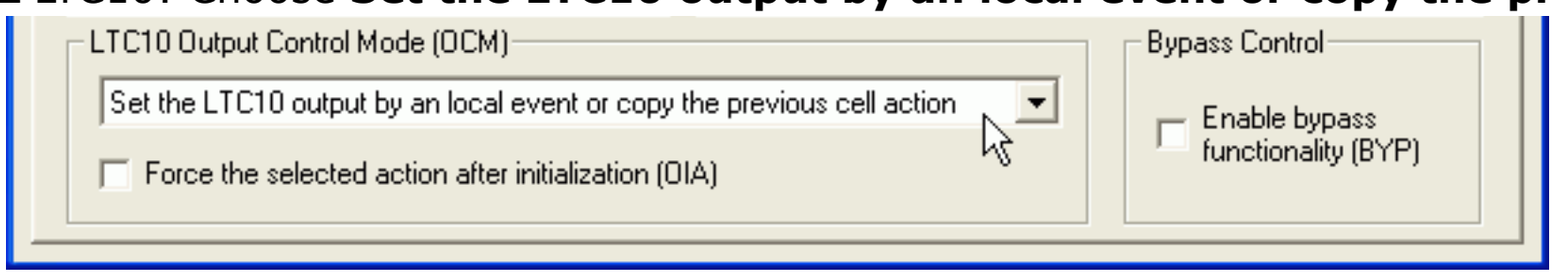
- LTC8: Choose **Reset the LTC8 output by an local event or copy the previous cell action**,



- LTC9: Choose **Reset the LTC9 output by an local event**



- LTC10: Choose **Set the LTC10 output by an local event or copy the previous cell action**.

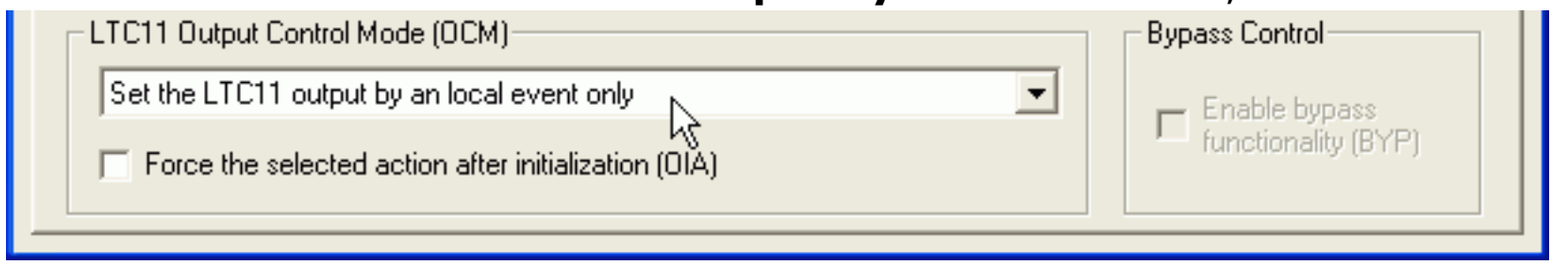


The high side (LTC7/LTC8) and the low side (LTC9/LTC10) of phase V are configured. Close the **LTC7** to **LTC10** dialogs.

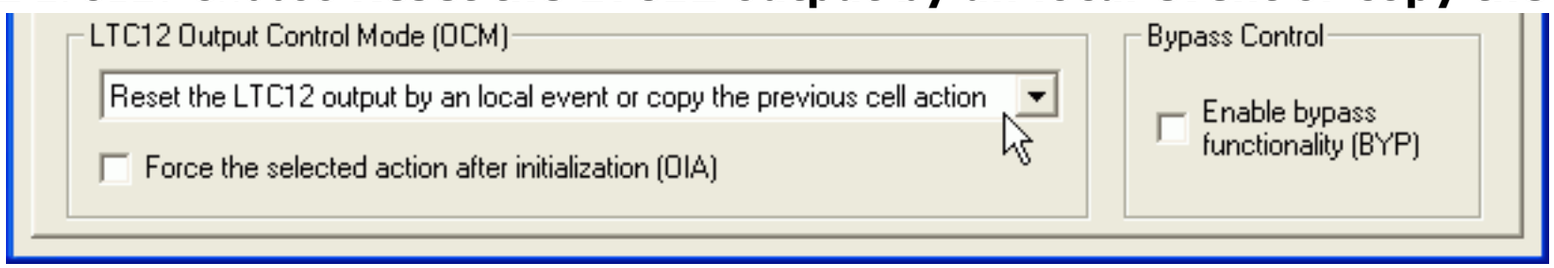
Exercise 7: PWM

For phase W open cell LTC11 to LTC14 and configure them also in compare mode:

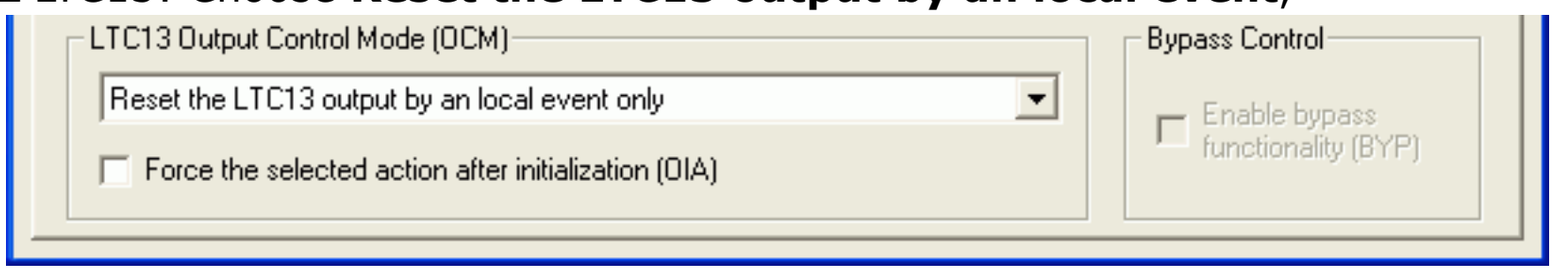
- LTC11: Choose **Set the LTC11 output by an local event**,



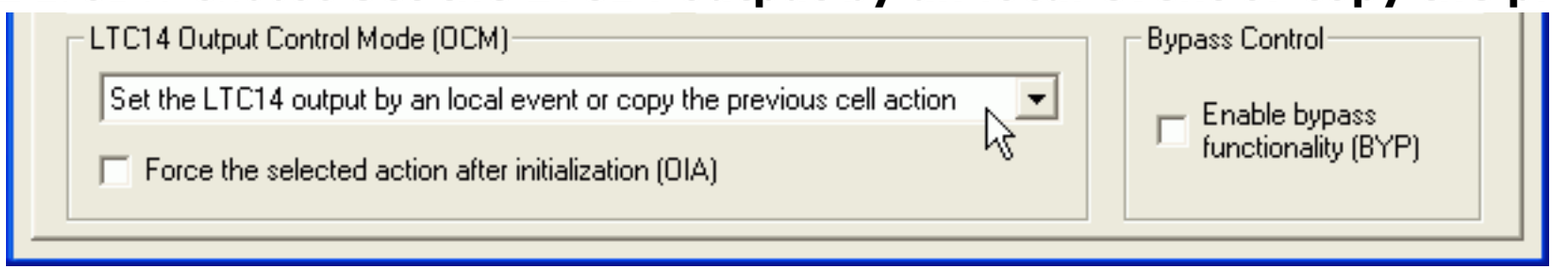
- LTC12: Choose **Reset the LTC12 output by an local event or copy the previous cell action**,



- LTC13: Choose **Reset the LTC13 output by an local event**,



- LTC14: Choose **Set the LTC14 output by an local event or copy the previous cell action**.

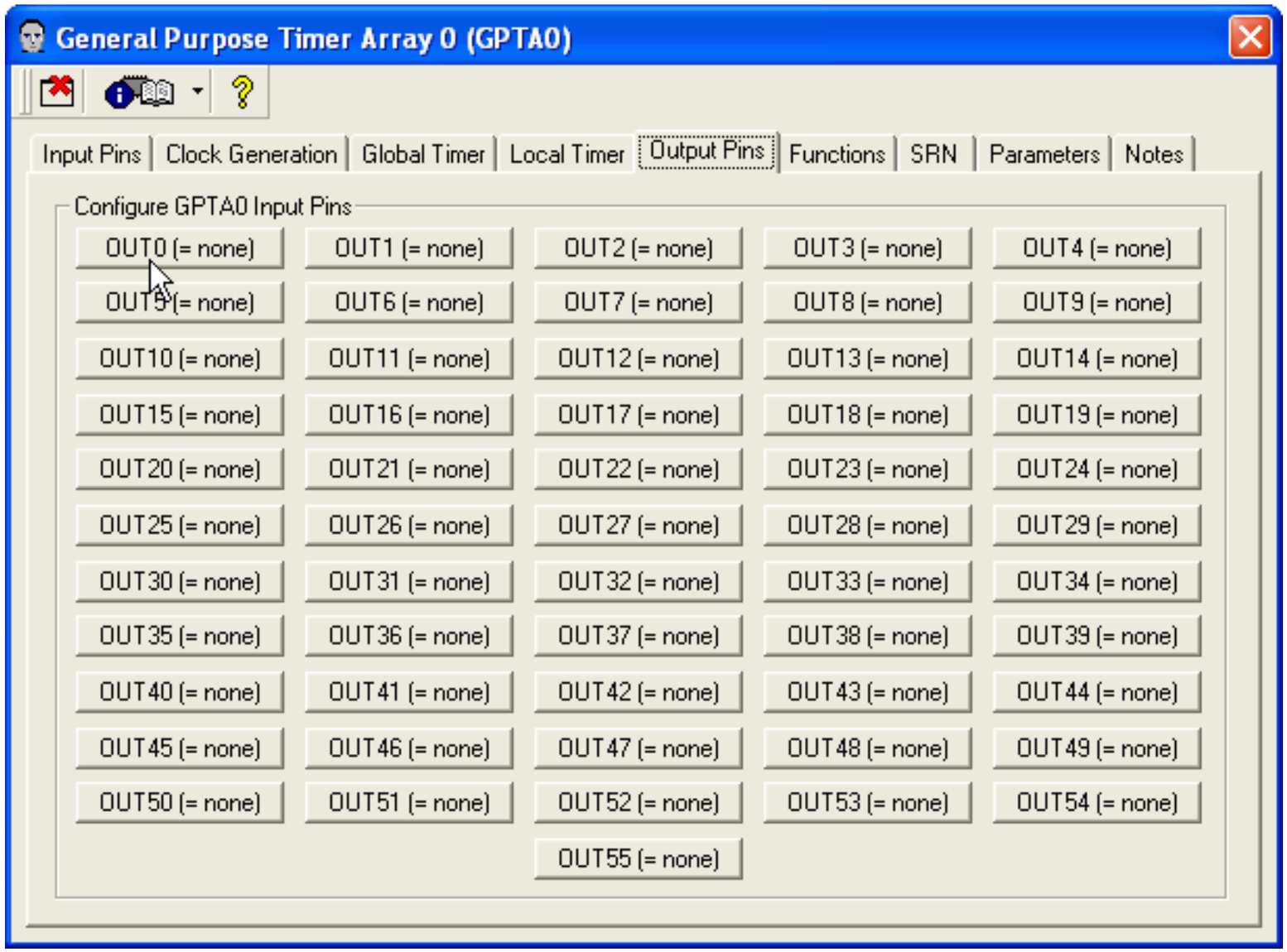


The high side (LTC11/LTC12) and the low side (LTC13/LTC10) of phase W are configured. Close the **LTC11** to **LTC14** dialogs.

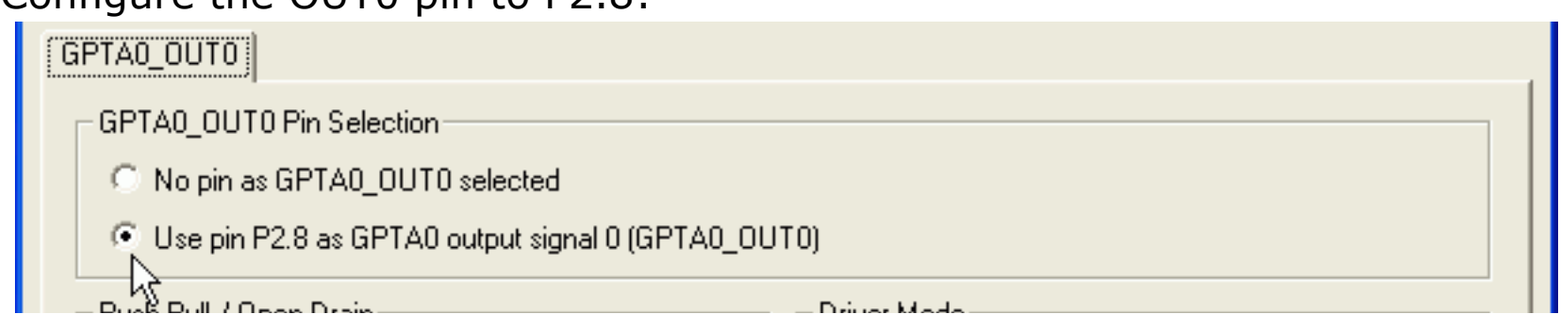
To connect the data out of the cells to the port the **Output Pins** has to be configured.

Exercise 7: PWM

On the **Output Pins** page open the **OUT0**, **OUT1**, **OUT8**, **OUT9**, **OUT10**, **OUT11** properties.

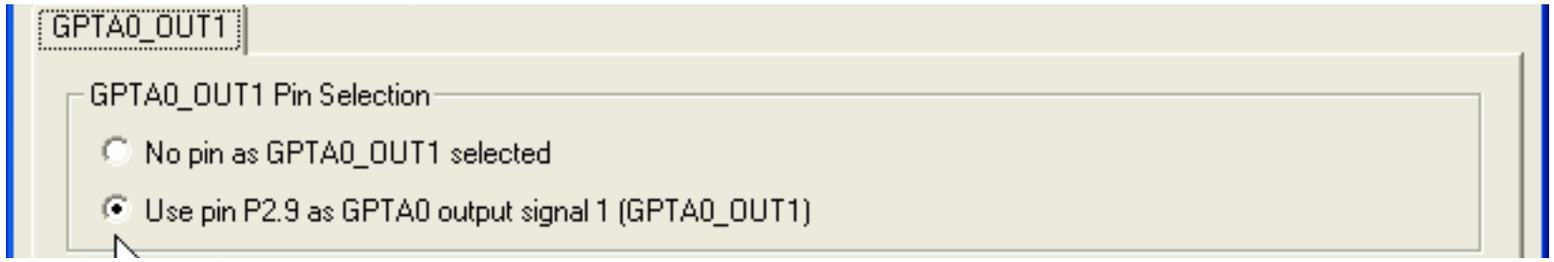


Configure the OUT0 pin to P2.8:

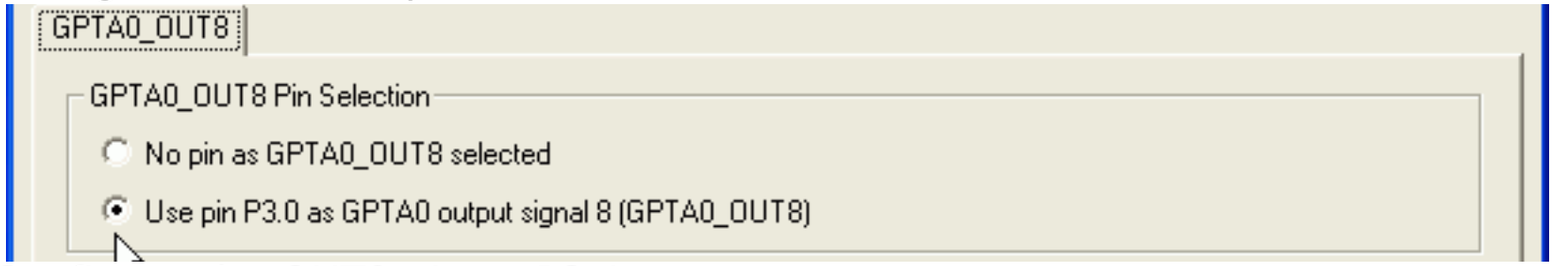


Exercise 7: PWM

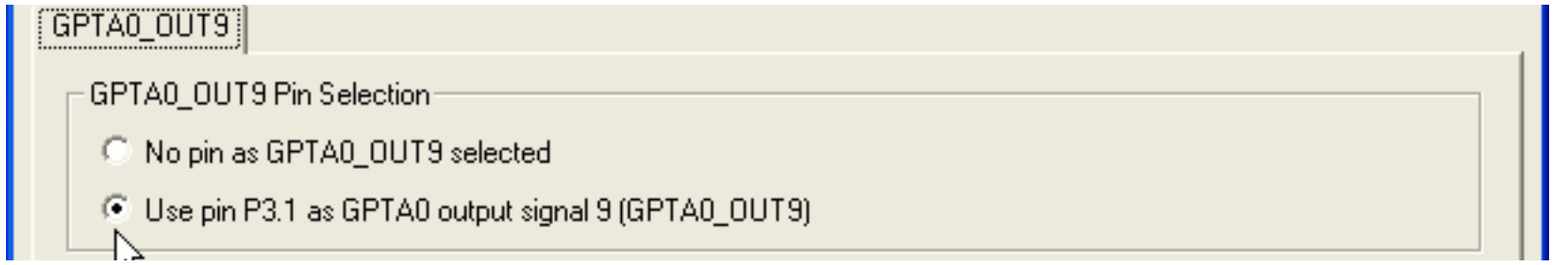
Configure the OUT1 pin to P2.9:



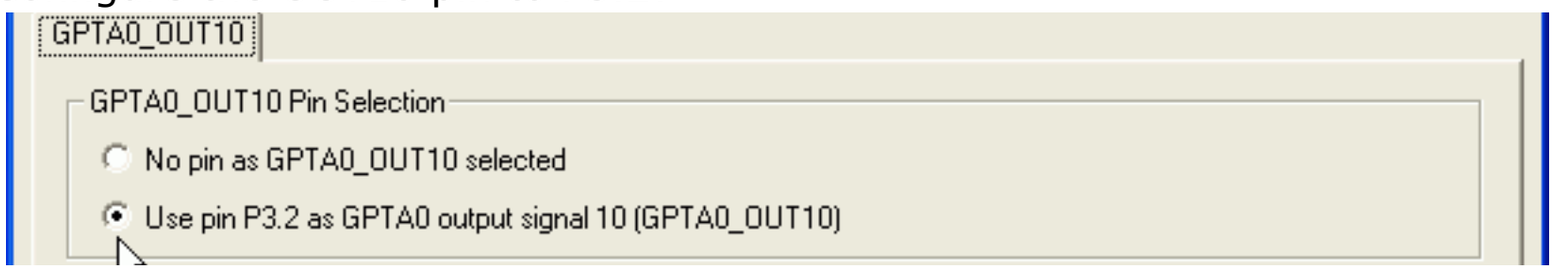
Configure the OUT8 pin to P3.0:



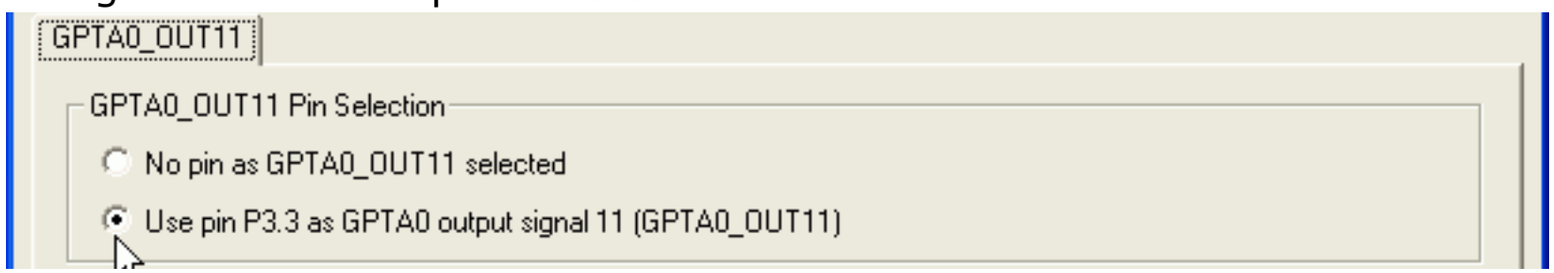
Configure the OUT9 pin to P3.1:



Configure the OUT10 pin to P3.2:

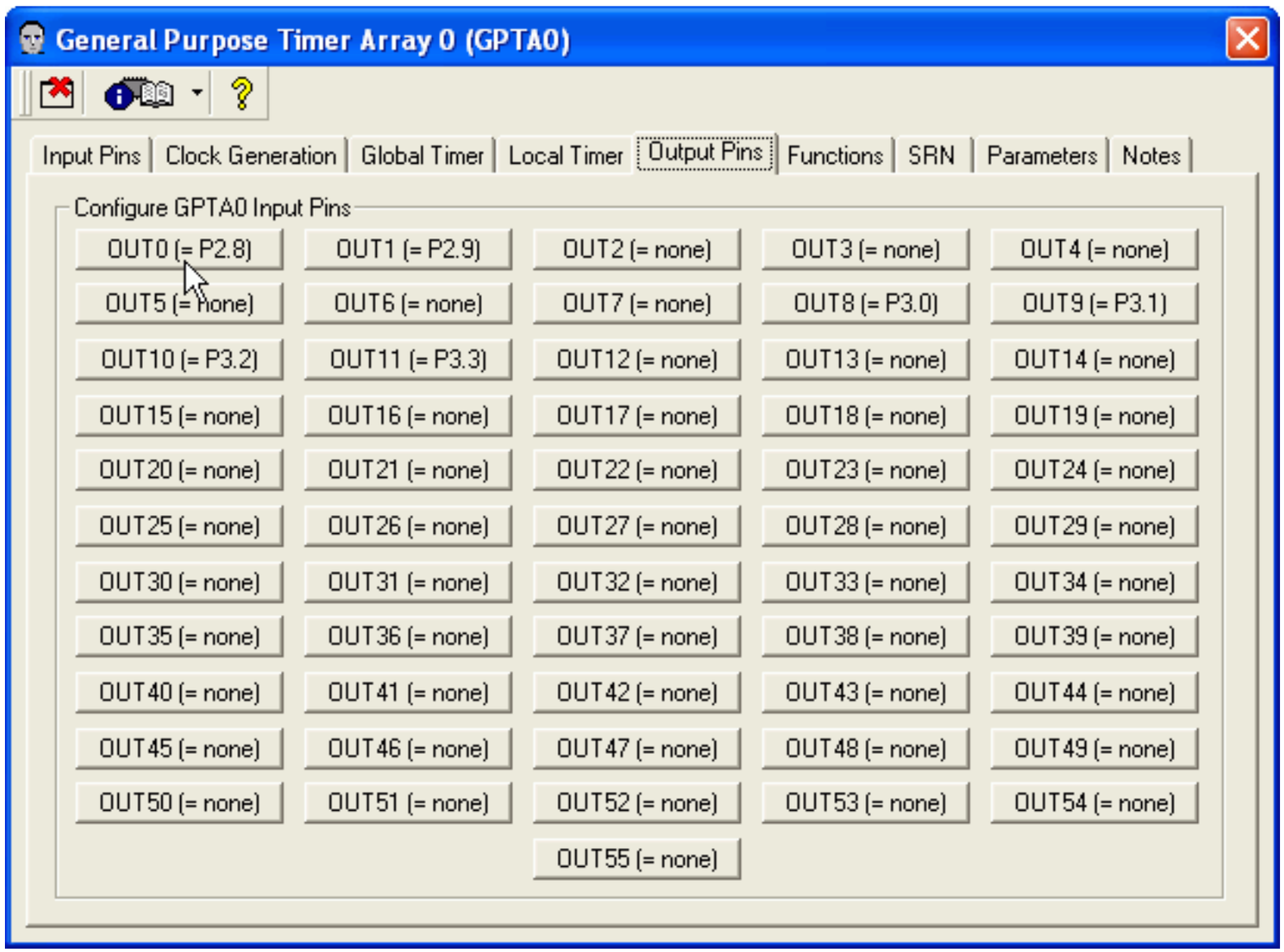


Configure the OUT11 pin to P3.3:



Exercise 7: PWM

Close the 6 **Configure Alternate Pin Functions** dialogs.
The **Output Pins** page displays the complete configuration of the output pins.

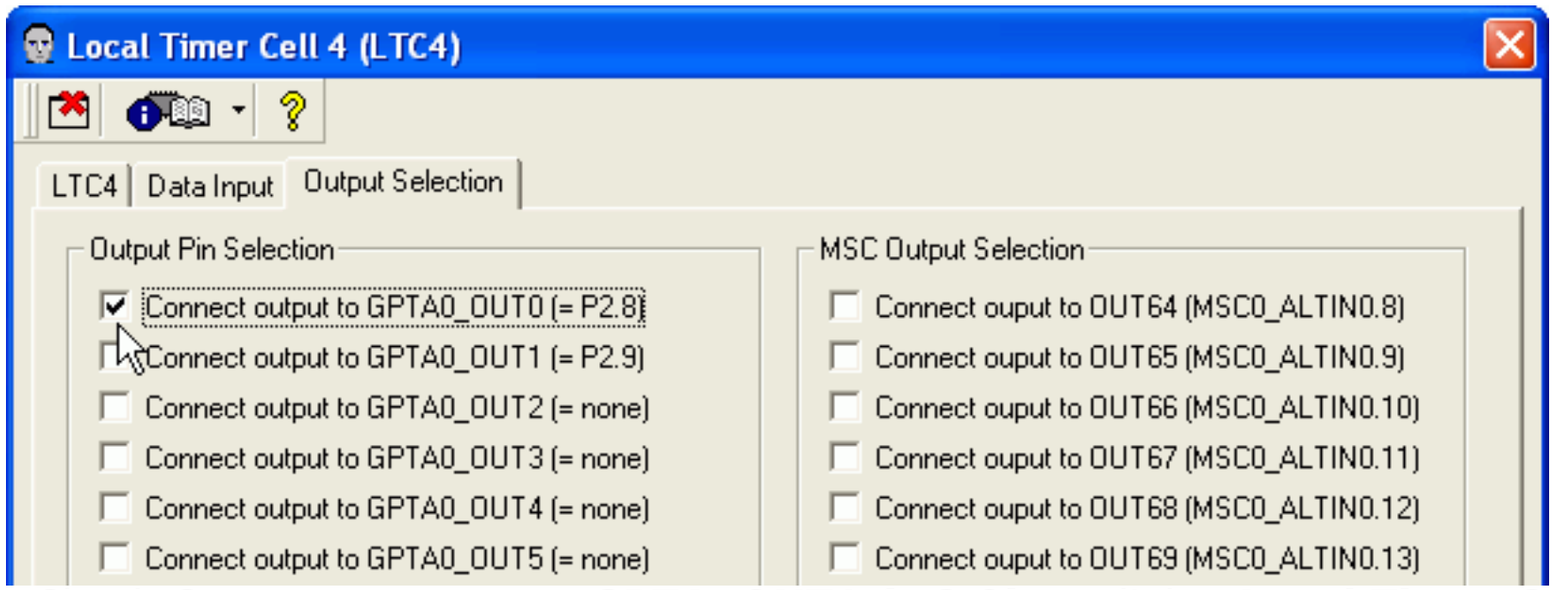


Select the **Local Timer** page again and click **LTC4**, **LTC6**, **LTC8**, **LTC10**, **LTC12**, **LTC14**.

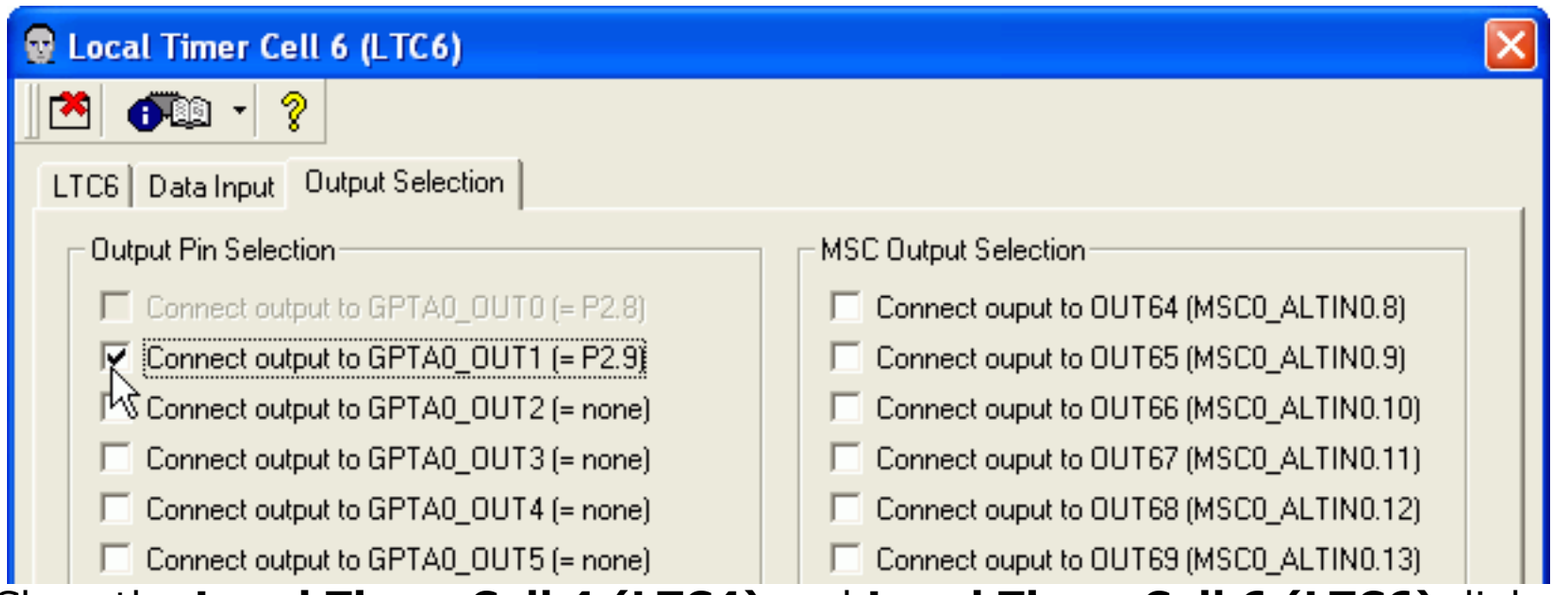
Exercise 7: PWM

On the **Output Selection** page of each dialog

■ Check **Connect output to GPTA_OUT0 (=2.8)** on dialog **Local Timer Cell 4** for the high side of phase U.



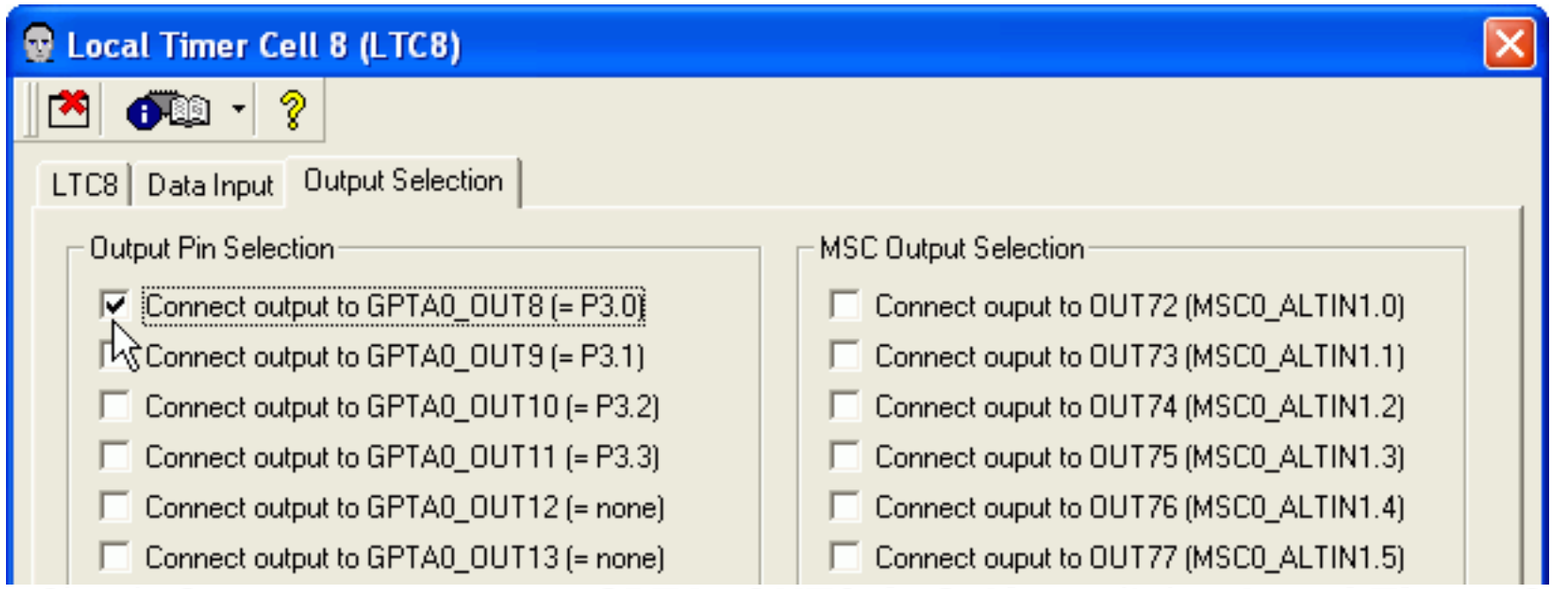
■ Check **Connect output to GPTA_OUT1 (=2.9)** on dialog **Local Timer Cell 6** for the low side of phase U.



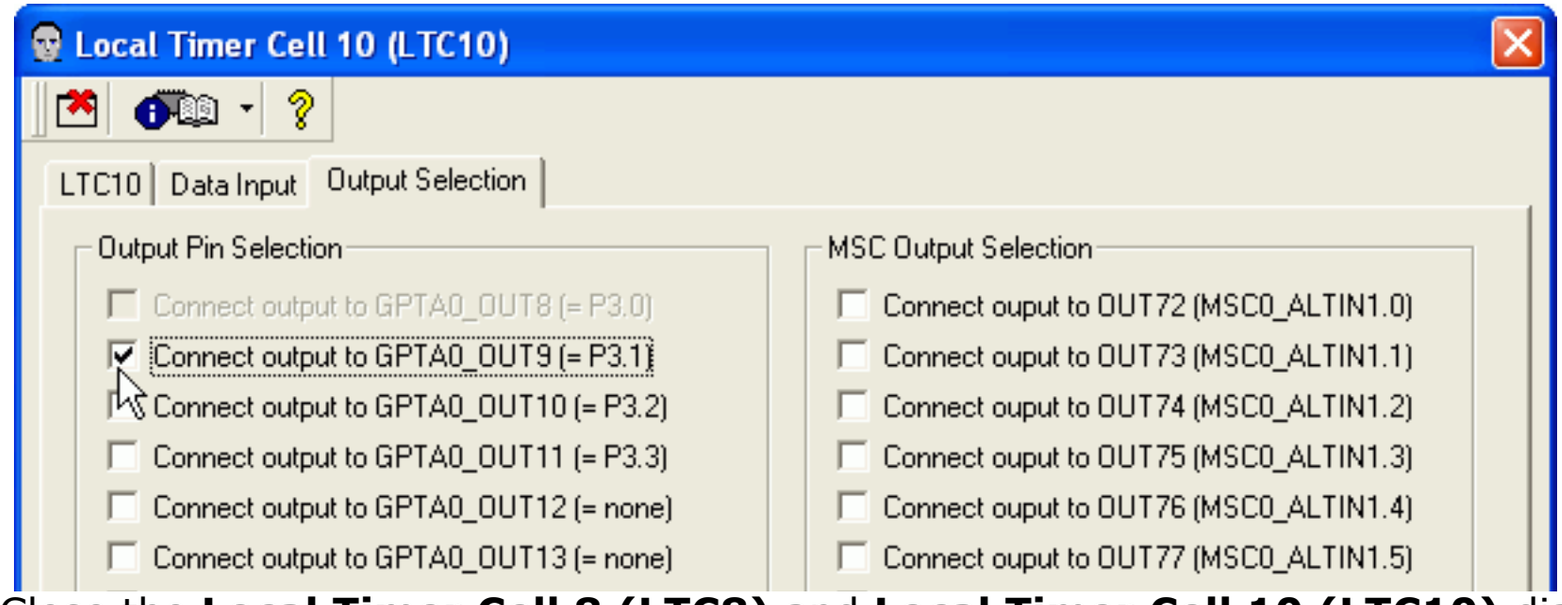
Close the **Local Timer Cell 4 (LTC4)** and **Local Timer Cell 6 (LTC6)** dialogs.

Exercise 7: PWM

■ Check **Connect output to GPTA_OUT8 (=3.0)** on dialog **Local Timer Cell 8** for the high side of phase V.



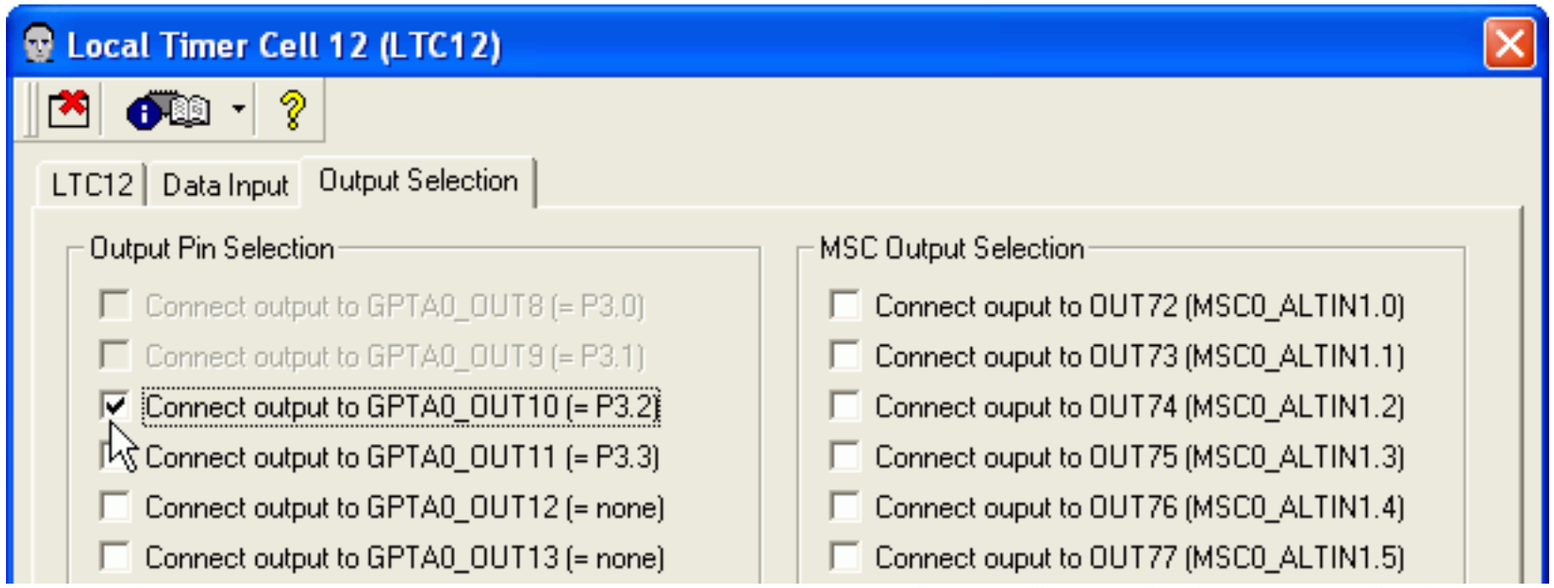
■ Check **Connect output to GPTA_OUT9 (=3.1)** on dialog **Local Timer Cell 10** for the low side of phase V.



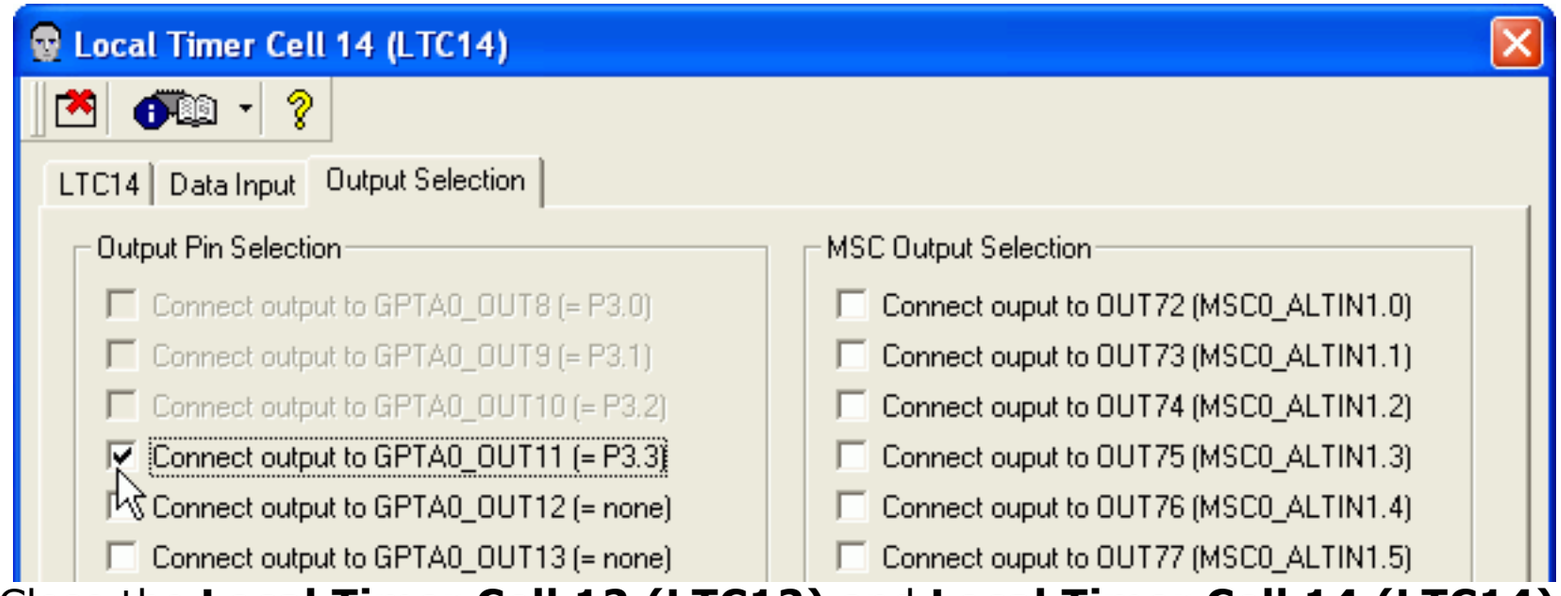
Close the **Local Timer Cell 8 (LTC8)** and **Local Timer Cell 10 (LTC10)** dialogs.

Exercise 7: PWM

■ Check **Connect output to GPTA_OUT10 (=3.2)** on dialog **Local Timer Cell 12** for the high side of phase W.



■ Check **Connect output to GPTA_OUT11 (=3.3)** on dialog **Local Timer Cell 14** for the low side of phase W.

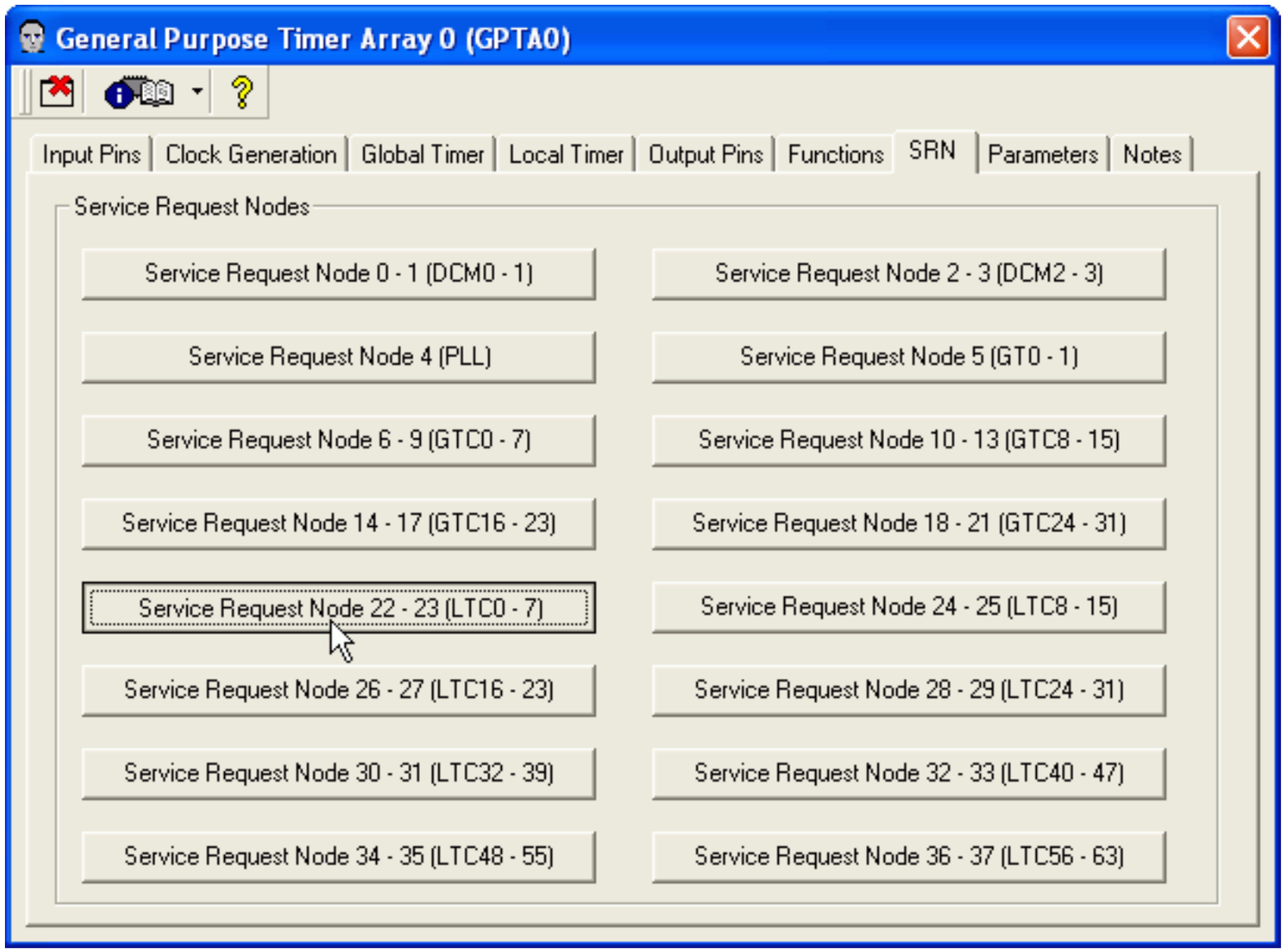


Close the **Local Timer Cell 12 (LTC12)** and **Local Timer Cell 14 (LTC14)** dialogs.

Exercise 7: PWM

7. Set up the GPTA0 properties. Service Request Node.

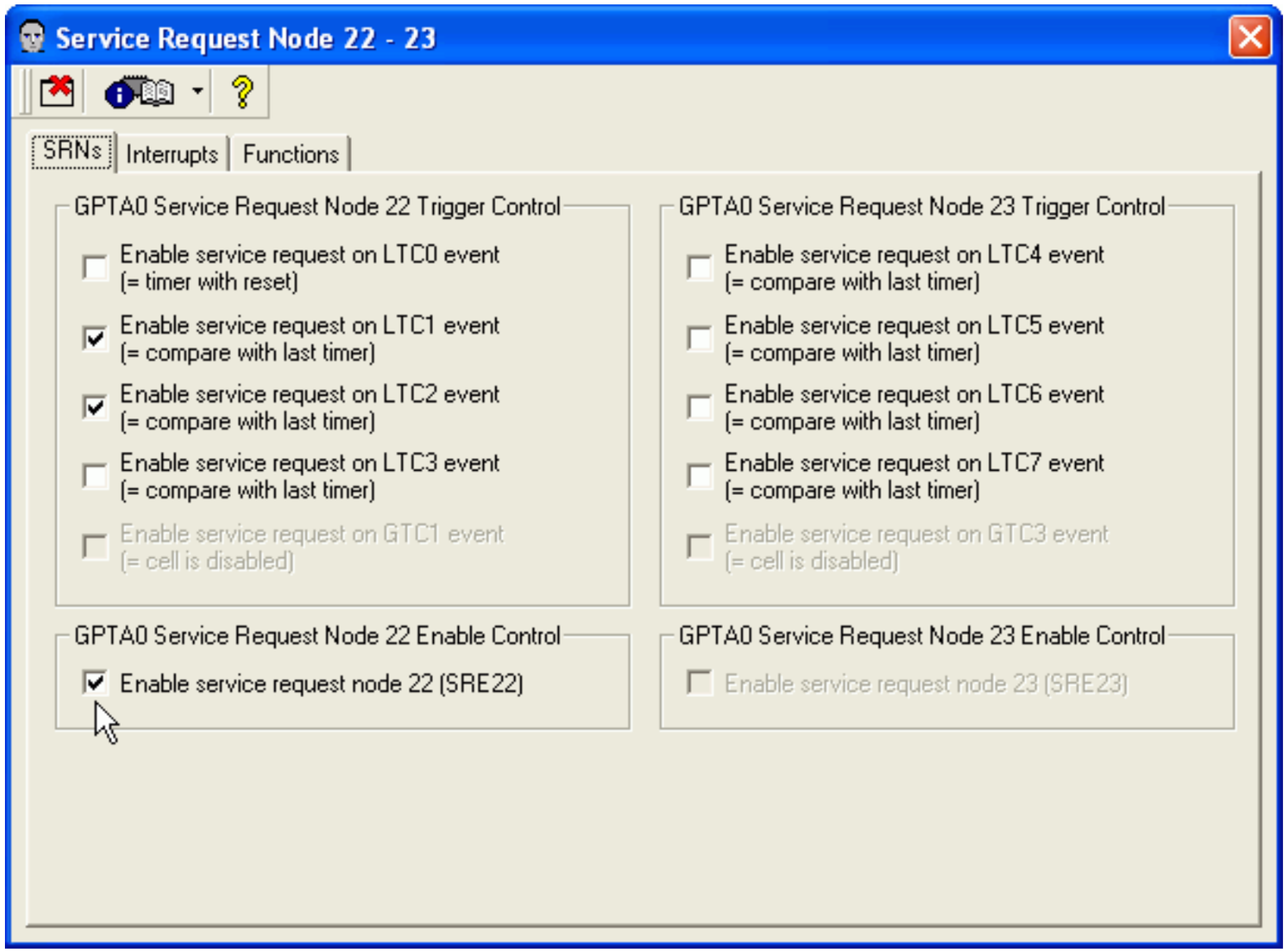
LTC1 and LTC2 are configured in period and mid period compare mode. To generate an interrupt on a compare match, the appropriate Service Request Node (SRN) has to be configured.
On the **SRN** page click **Service Request Node 22-23 (LTC0-7)**.



Exercise 7: PWM

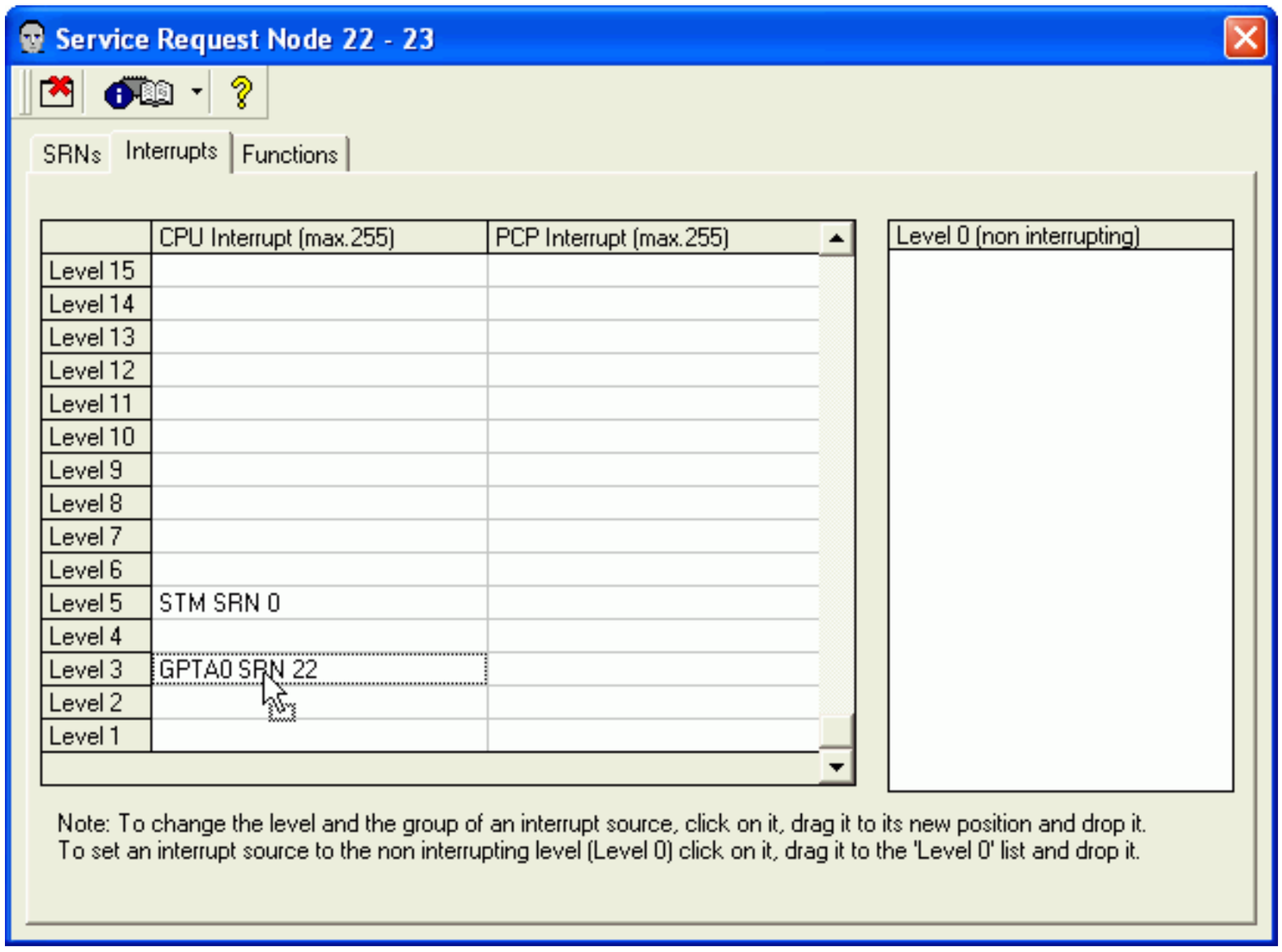
On the **SRNs** page enable the interrupt generation on the period and on the mid period compare match.

- Check **Enable service request on LTC1 event (=compare with last timer)**,
- Check **Enable service request on LTC2 event (=compare with last timer)**,
- Check **Enable service request node 22 (SRE22)**.



Exercise 7: PWM

On the **Interrupts** page and drag **GPTA0 SRN 22** from Level 0 to the CPU Interrupt level 3.

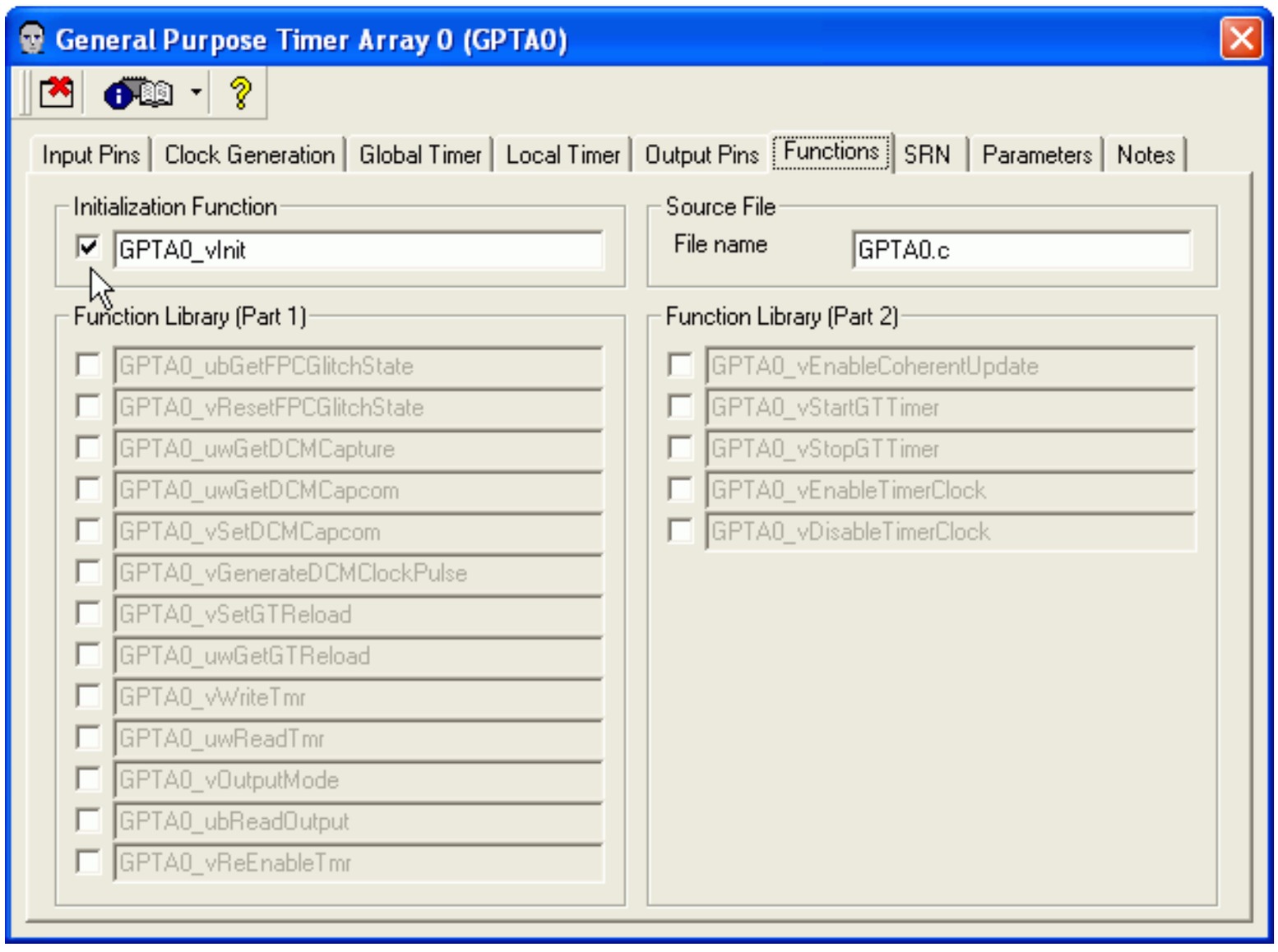


Close the **Service Request Node 22-23** dialog.

Exercise 7: PWM

On the **Functions** page

■ Check the `GPTA0_vInit` function.

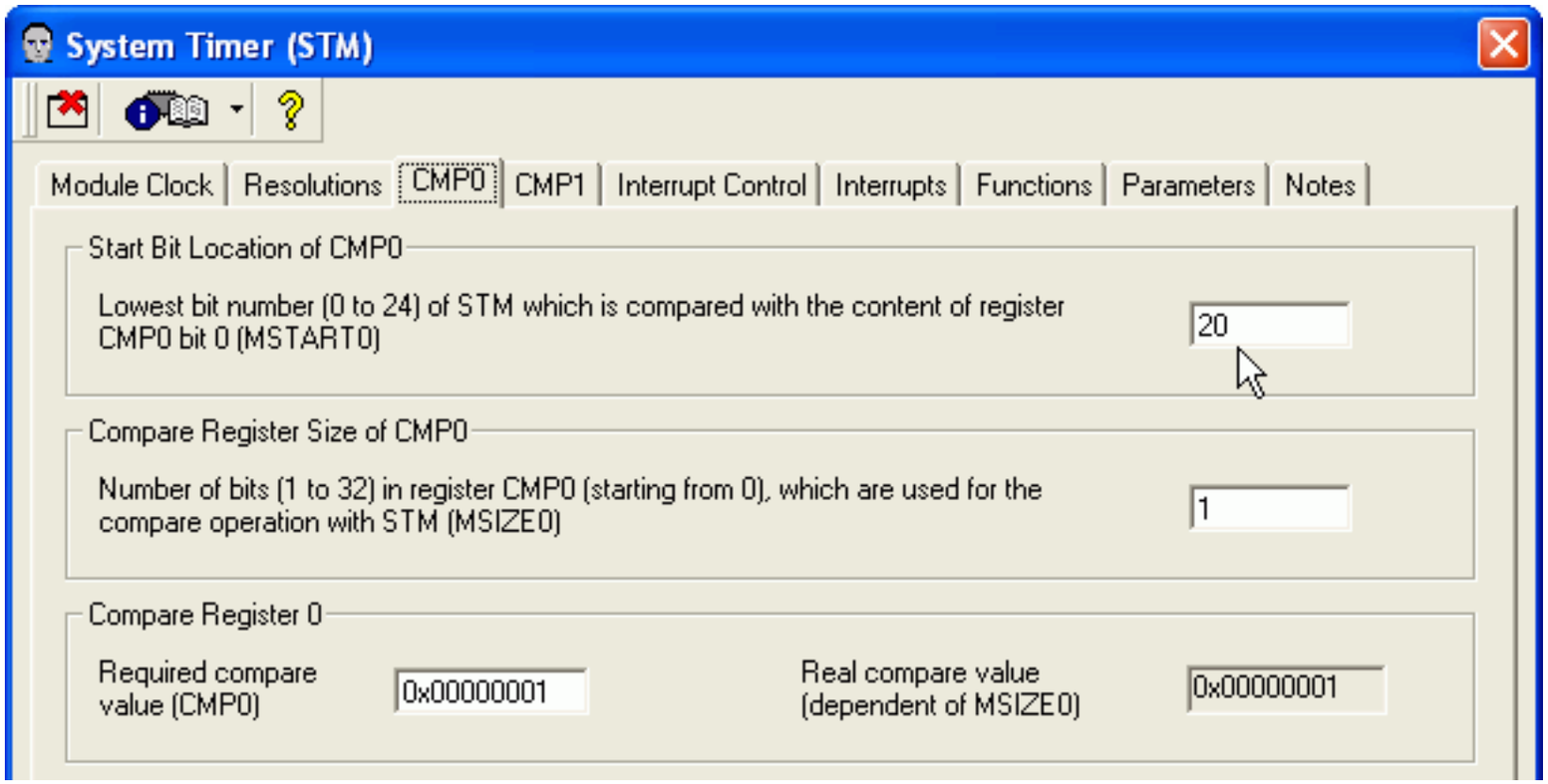


The GPTA is completely configured to generate a complementary 3-phase PWM signal. Click the **Close** icon on the dialog toolbar to close the **General Purpose Timer Array 0 (GPTA0)** dialog.

8. Modify the STM properties.

Open the System timer properties and select the **CMP0** page

- Change the **Start Bit Location of CMP0** to 20



The system timer interrupt is used to update the PWM values.

9. Generate the application framework

Click on the **Generate Code** icon  on the application toolbar to start the code generation process. Save and close the *DAvE* project.

10. Add a new project to the Tasking Workspace

Open the *Tasking* EDE. Choose **File > Configure Project Space... > Add new project** and add a new project `c:\infineon\pwm\pwm.pjt`.

Exercise 7: PWM

11. Add the application framework

In the **Project Properties** dialog click the **Scan** icon  . A dialog appears. In the Pattern field, enter `*.c;*.h`. This will select all generated files of the application framework. Select the project directory and click **OK**.

12. Set current project

Use the context menu in the workspace window to make the pwm project the current project.

13. Load the project options

Choose **Project > Load Options**. In the **Filename** field enter `c:\infineon\tc1796_intmem.opt`.

14. Build the application

Click the **Build** icon  on the Build toolbar. The Build process finishes successfully.

15. Add the user code

Add the following code to `GPTA0.c`. Include the standard math library at (GPTA0_General,2).

```
// USER CODE BEGIN (GPTA0_General,2)
#include <math.h>
// USER CODE END
```

Define the following constants at (GPTA0_General,4).

```
// USER CODE BEGIN (GPTA0_General,4)

#define LTC_FREQ                37500000  // = 37.5MHz
#define PWM_FREQ                20000     // = 20.0kHz
#define DEADTIME                1E-6      // = 1µs

#define DEADTIME_CNTS           ((short) (LTC_FREQ * DEADTIME))
#define PWM_PERIOD_CENTER_CNTS  ((short) ((LTC_FREQ / PWM_FREQ) / 2)) // same as GPTA0_LTCXR02

// USER CODE END
```

Exercise 7: PWM

Declare the following global variables at (GPTA0_General,7). For the sinus a look up table is used to save time. Three circular buffers pointers are used to step through the elements of the array. Incrementing a circular buffer pointer that points to the last element results in a pointer to the first element.

```
// USER CODE BEGIN (GPTA0_General,7)
#define SINE_STEPS 360
short  __near __circ duty[SINE_STEPS];  // duty lookup table in a circular buffer
short  __near __circ *dutyU, *dutyV, *dutyW;
const float pi = 3.14159;
// USER CODE END
```

In the `GPTA_vInit` function disable the interrupt for LTC1 and LTC2 at (Init, 3). The PWM will be started by the first STM interrupt.

```
// USER CODE BEGIN (Init,3)
GPTA0_LTCCTR01_REN = 0;    // disable period interrupt
GPTA0_LTCCTR02_REN = 0;    // disable mid period interrupt
// USER CODE END
```

Fill the duty lookup table and initialize the circular buffer pointer at (Init, 4)

```
// USER CODE BEGIN (Init,4)
for(int i=0;i<SINE_STEPS;i++)
  duty[i] =  (PWM_PERIOD_CENTER_CNTS + 1) * 0.5*(sinf(i * 2*pi/SINE_STEPS) + 1);

dutyU = &duty[0];                // set to sin(0°)
dutyV = &duty[SINE_STEPS/3];     // set to sin(120°)
dutyW = &duty[-SINE_STEPS/3];    // set to sin(240°)
// USER CODE END
```

Exercise 7: PWM

Configure the period and mid period interrupts at `GPTA0_viSRN22` in file `GPTA.c`. Add the code for the period and mid period interrupt at (SRN22, 2). At zero percent duty the low side must be set to active, i.e. $t_{on,L}$ should be ignored. This is done by setting $t_{on,L}$ to -2 (=$FFFE_H$), a value that is never reached.

```
// USER CODE BEGIN (SRN22,2)
unsigned int n;
if (GPTA0_LTCCTR01_REN)
{
  GPTA0_LTCCTR01_REN = 0;  // disable period interrupt
  STM_ICR_CMP0EN = 1;  // enable the stm interrupt

  //  update duty U
  n = PWM_PERIOD_CENTER_CNTS + *dutyU;
  GPTA0_LTCXR04 = n;
  GPTA0_LTCXR06 = n + DEADTIME_CNTS;
  //update duty V
  n = PWM_PERIOD_CENTER_CNTS + *dutyV;
  GPTA0_LTCXR08 = n;
  GPTA0_LTCXR10 = n + DEADTIME_CNTS;
  //update duty W
  n = PWM_PERIOD_CENTER_CNTS + *dutyW;
  GPTA0_LTCXR12 = n;
  GPTA0_LTCXR14 = n + DEADTIME_CNTS;

}
...
```

Exercise 7: PWM

```
else
{
  GPTA0_LTCCTR02_REN = 0;  // disable mid period interrupt
  GPTA0_LTCCTR01_REN = 1;  // enable  period interrupt

  // update duty U
  n = PWM_PERIOD_CENTER_CNTS - *dutyU;
  GPTA0_LTCXR03 = n;
  GPTA0_LTCXR05 = (n == PWM_PERIOD_CENTER_CNTS) ? 0xfffe :
                          ((n < DEADTIME_CNTS) ? 0xffff : n - DEADTIME_CNTS);
  // update duty V
  n = PWM_PERIOD_CENTER_CNTS - *dutyV;
  GPTA0_LTCXR07 = n;
  GPTA0_LTCXR09 = (n == PWM_PERIOD_CENTER_CNTS) ? 0xfffe :
                          ((n < DEADTIME_CNTS) ? 0xffff : n - DEADTIME_CNTS);
  // update duty W
  n = PWM_PERIOD_CENTER_CNTS -  *dutyW;
  GPTA0_LTCXR11 = n;
  GPTA0_LTCXR13 = (n == PWM_PERIOD_CENTER_CNTS) ? 0xfffe :
                          ((n < DEADTIME_CNTS) ? 0xffff : n - DEADTIME_CNTS);
  }
  // USER CODE END
```

Exercise 7: PWM

Enable the mid period interrupt in file `MAIN.c` before the forever loop at (Main,9):

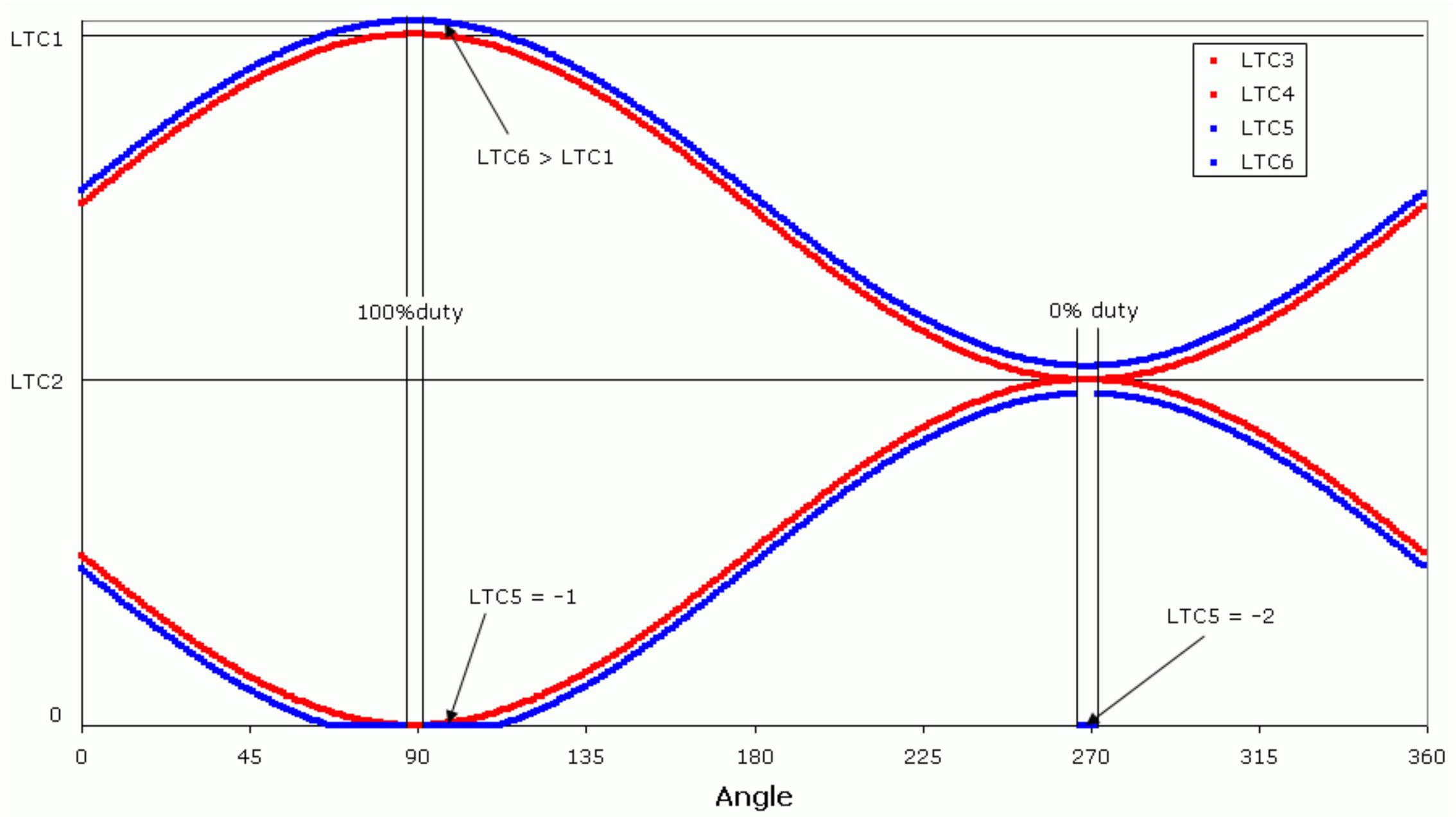
```
// USER CODE BEGIN (Main,9)
STM_ICR_CMP0EN = 1;  // enable the stm interrupt
for(;;)  // forever
    ;
// USER CODE END
```

In file `STM.c` define the `*dutyU, *dutyV, *dutyW` as `extern` and add to function `STM_viSRN0` (SRN0,3):

```
// USER CODE BEGIN (SRN0,3)
STM_ICR_CMP0EN = 0;          // disable the stm interrupt
GPTA0_LTCCTR02_REN = 1;      // enable mid period interrupt
dutyU++; dutyV++; dutyW++;   // circular step through the sinus lookup table
// USER CODE END
```

Exercise 7: PWM

At 0% duty LTC3 and LTC4 have the same compare value and the output is following the action request of the higher cell LTC4, e.g. the high side is set to passive. LTC5 is -2 and therefor ignored, so that the low side is set to active.
At 100% duty LTC5 has a compare value of -1 and is set immediately. LTC6 is larger than the period in LTC1, so that a reset is never happen. I.e. The low side is set to passive.



Exercise 7: PWM

16. Build the application

Click the **Build** icon  on the Build toolbar. The Build process finishes successfully.

17. Debug the application

Click the **Debug** icon  on the Build toolbar to open the *CrossView Pro* debugger.

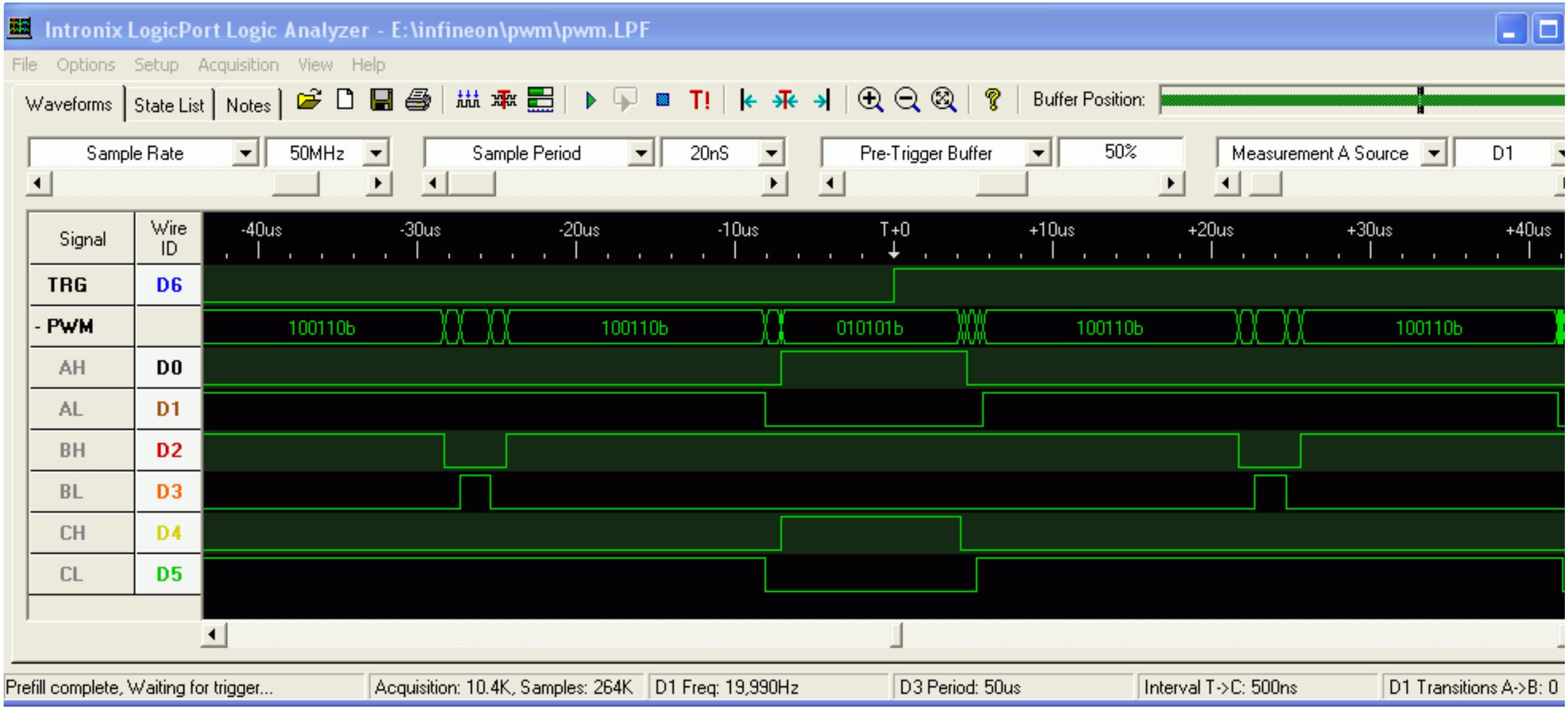
18. Connect an Oscilloscope

Connect the pins P2.8, P2.9, P3.0, P3.1, P3.2, P3.3 to a logic analyzer. Connect P2.10 to the trigger input.

Exercise 7: PWM

19. Run the application

Click the **Run** icon ▶ on the *CrossView Pro* toolbar and see the logic analyzer output.
The cursor is adjusted to measure the dead time.

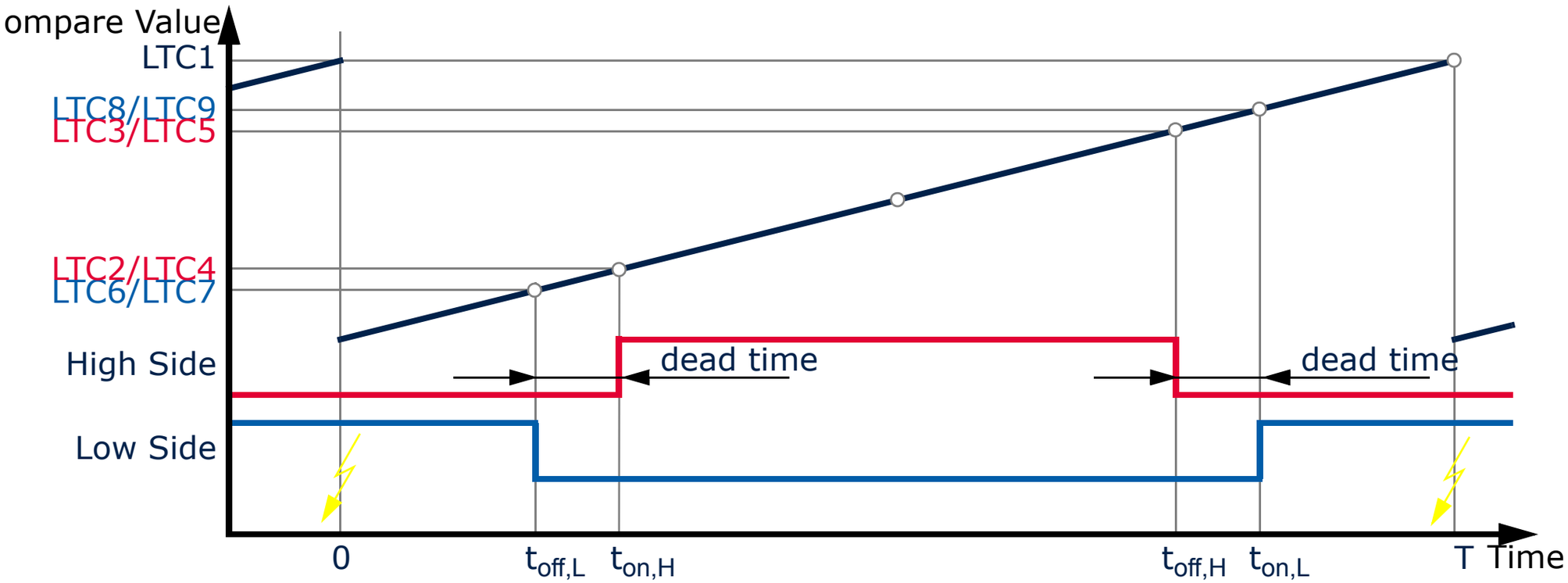


Exercise 7: PWM

PWM with coherent update

The exercise above uses 2 interrupts to set-up the t_{on} and t_{off} values of the low and high side. This system setup might be critical when a interrupt is delayed by more than half a period and the interrupt routine set the new values in the deadtime period. Then only one side will be set to a new value and this might cause a short.
Therefor in systems with heavy interrupt load another configuration is needed. Instead of using shadow registers which are not available within the LTC a second set of LTCs can be set-up and the reset timer switches by HW from on set to the other. This is called coherent update (CUD). The exercise solutions contains a project pwm_cud which implements such a solution. This solution uses twice as much LTCs for each PWM, so that for a complementary 3-phase PWM 26 cells are used instead of 15.

Exercise 7: PWM



	Cell	SOL	SOH	Output Control Mode	Port
Reset Timer	LTC0		-	Toggle select line (SL) by HW on reset, Interrupt on reset to set-up $t_{on,H}$,$t_{off,H}$,$t_{on,L}$,$t_{on,L}$	-
Compare Period	LTC1	1	1	-	-
Compare High Side On	LTC2	1	0	Set output by a local event	-
Compare High Side Off	LTC3	1	0	Reset output by a local event or copy the previous cell action	-
Compare High Side On	LTC4	0	1	Set output by a local event or copy the previous cell action	-
Compare High Side Off	LTC5	0	1	Reset output by a local event or copy the previous cell action	P2.8
Compare Low Side Off	LTC6	1	0	Reset output by a local event	-
Compare Low Side On	LTC7	1	0	Set output by a local event or copy the previous cell action	-
Compare Low Side Off	LTC4	0	1	Reset output by a local event or copy the previous cell action	-
Compare Low Side On	LTC5	0	1	Set output by a local event or copy the previous cell action	P3.0