Oscillator Configuration tool

2014-11-12

1 Contents

2 Purpose of the tool

The purpose of this tool is to provide, for users of the mE/PIC compilers, the necessary ***configuration settings*** and, if necessary, the **register values** for a **wanted oscillator configuration**. Additionally it allows to see the clock frequencies along the chosen clock path, and eventually the actual CPU clock (to enter in the Project Settings/Edit Project) and the used USB clock (if any).

A lot of problems with PIC projects are due to wrong choices in the oscillator configuration, especially for the more complex PIC's (e.g. those with USB).

The tool provides a visual interface. It shows the oscillator block diagram of a PIC(family), and the user can choose which oscillator type is going to be used, which signal paths are active, which choices are made (e.g. if the PLL is active), etc...

Out of that, the tool decides what the configuration bits will be and what the initialisation code (register settings) needs to be.

3 Installation of the tool

This is very simple: just extract the "Oscillator_Configuration.zip" file to a dedicated directory, start up by executing "Oscillator_Configuration.exe" and select a PIC.

The tool uses the following files:

- The .mlk file of the processor selected PIC. The .mlk file(s) are part of a compiler's installation. The paths to those .mlk files are defined in the file "Oscillator_Configuration.ini". Initially this file is absent, it is created when the tool is started up the first time. If the tool does not find the .mlk file of a PIC then these paths can be adapted manually (the latter is only needed if the compiler is installed in a non standard path).
- A bitmap file per PIC family, e.g. "Oscillator_P18F27J53.bmp", showing the oscillator block diagram (with perhaps slight modifications) of the PIC(family) datasheet. This .bmp file has to be created for every new PIC(family) to be added to the tool's capabilities. Delivered with the tool.
- A text file per PIC family, e.g. "Oscillator_P18F27J53_lines.txt", which specifies where the items are to be handled in the visual interface, and how the output (configuration bits and register settings) are to be created. This .txt file has to be created for every new PIC(family) to be added to the tool's capabilities. Delivered with the tool.
- A text file containing all supported PIC's with the name of the bitmap and .txt files they use, delivered with the tool.

4 Usage of the tool

4.1 Overview

- Start the tool and select a PIC (section 4.2 for details).
- Select the necessary oscillator types and clock paths in the oscillator block diagram (section 4.4 for details).
- Copy (=select) the Configuration bits outcome (visible at top right in the tool) in the Compiler's IDE "Edit Project " menu (section 4.5 for details).
- Copy the register settings (to be found in the "Code" menu) to your project's initialisation code (you can leave out the "default" code if you want) using the Clip Board functions (section 4.6 for details).
- If you want to see the clock frequencies along the clock path (especially the actual CPU clock, and/or the USB clock), then also fill in the frequency of the oscillator(s) (section 4.7 for details).

4.2 The menu-items

4.2.1 PIC

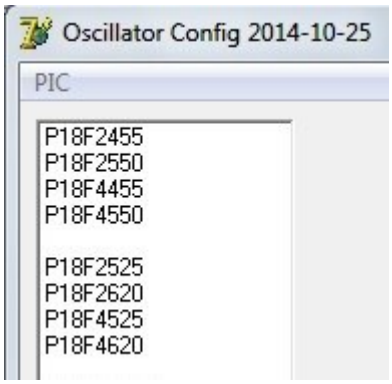


Figure 1

(only small part shown)

Used to select a PIC. The Pic's visible in the list are defined in the file "Oscillator_Configuration_PICs.txt", and of course, for those PIC's the necessary files (.bmp, .lines.txt) should be available. After selecting a PIC the main screen with the oscillator block diagram and the selectable/selected lines is shown.

4.2.2 Code

This is the code generated according the "register type" selections made in the oscillator block diagram (see Figure 2). The code may contain, in the line comments, the sentence part "(default"). The latter means that this is the state after a PIC reset. So, if the code is only used at startup of the program, then those "default" lines can be left out. See section 4.6 also.

4.2.3 Clear

Here 4 options are available:

- **Clear all**: All current selections (both config bits and register selections) are cleared, nothing is selected.
- **Clear Config bits**: only the selections related to configuration bits are cleared.
- **Clear Register Settings**: only selections related to register settings are cleared.
- **Reload All**: the initial settings belonging to the currently selected PIC are reloaded (both config bits and register settings).

4.2.4 Help

In the help file a list of the most important steps is presented about which items to select in the oscillator block diagram. See next section(s).

4.3 Starting up

First, select a PIC (in the menu, see section 4.2.1). This is done in the “PIC” menu.

Next, the main screen is shown (PIC dependent of course):

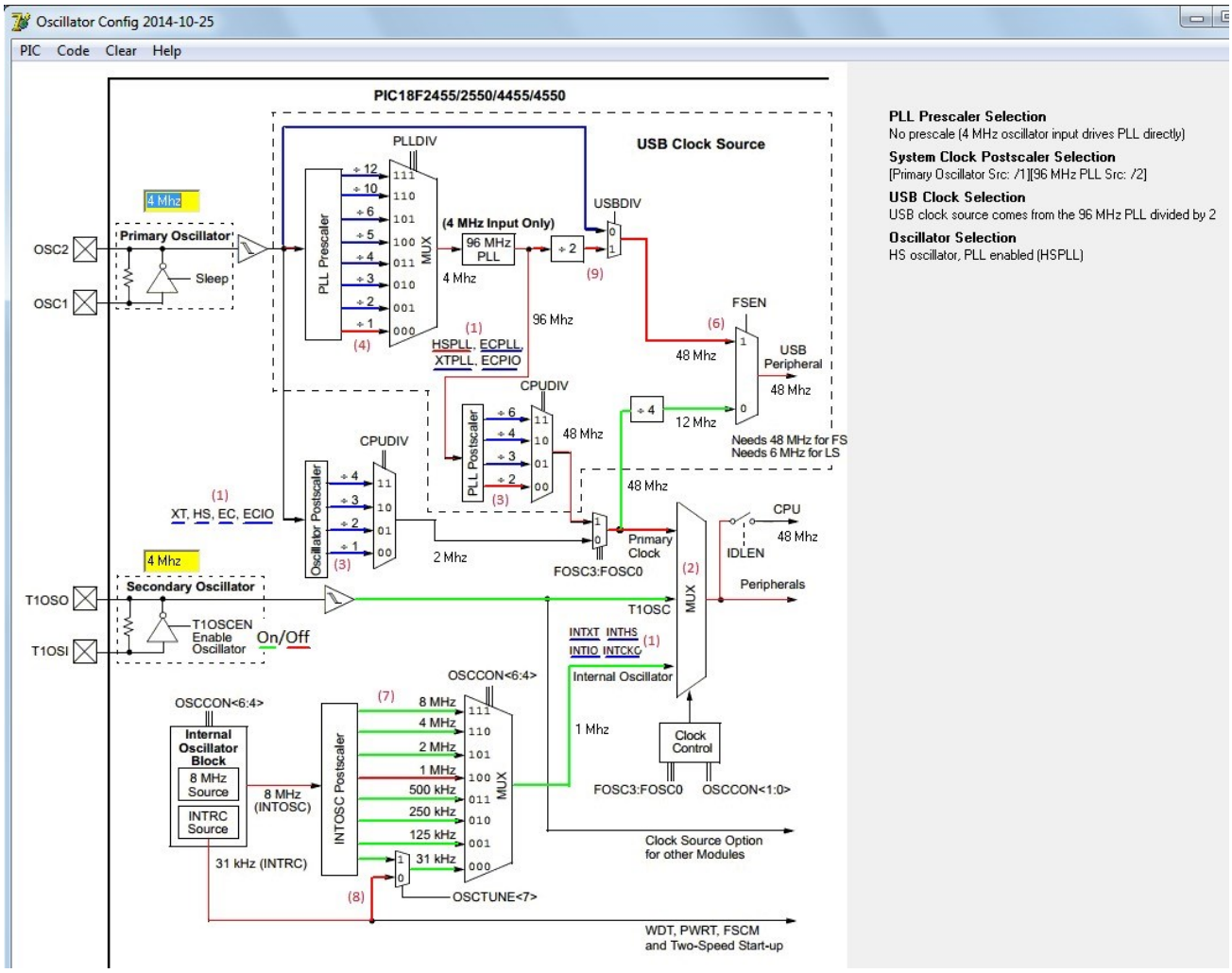


Figure 2

A “default” setting is shown: the oscillator configuration defined by config bits is defined by the .txt file contents, the oscillator configuration defined by register settings is (as much as possible) equal to the reset state of the PIC.

The main screen shows a number of features:

- Frequency entries: One or more fields allow to fill in a frequency. These are (normally) the frequencies of the crystals of the primary and/or secondary oscillators.
- Thick **blue** lines: these are selectable lines that will define configuration bits one selected. The configuration settings to select in the compiler’s IDE is shown at the top right hand of the tool.
- Thick **green** lines: these are selectable lines that will define the initialisation code regarding the oscillator in your project once selected. This init code can be made visible with the “Code” menu. It can be copied with ctrl-A/crtl-C to your project.
- Thick **red** lines: these are currently selected lines of one of the two types above. They will define configuration bits and/or register settings.
- Thin red lines: these are not selectable, but follow other selections to show a complete clock path, in addition to the selected parts.

4.4 Selecting the Oscillator Configuration

4.4.1 General Remarks

A selection is always made by clicking a (blue or green) line. Lines can not be unselected by clicking, only selecting another (mutual exclusive or group) line can achieve this.

The bracketed numbers – e.g. (1) - used in the list above can also be found (if appropriate) in the oscillator block diagram – in red- , see Figure 2 for the PIC18F2550 family.

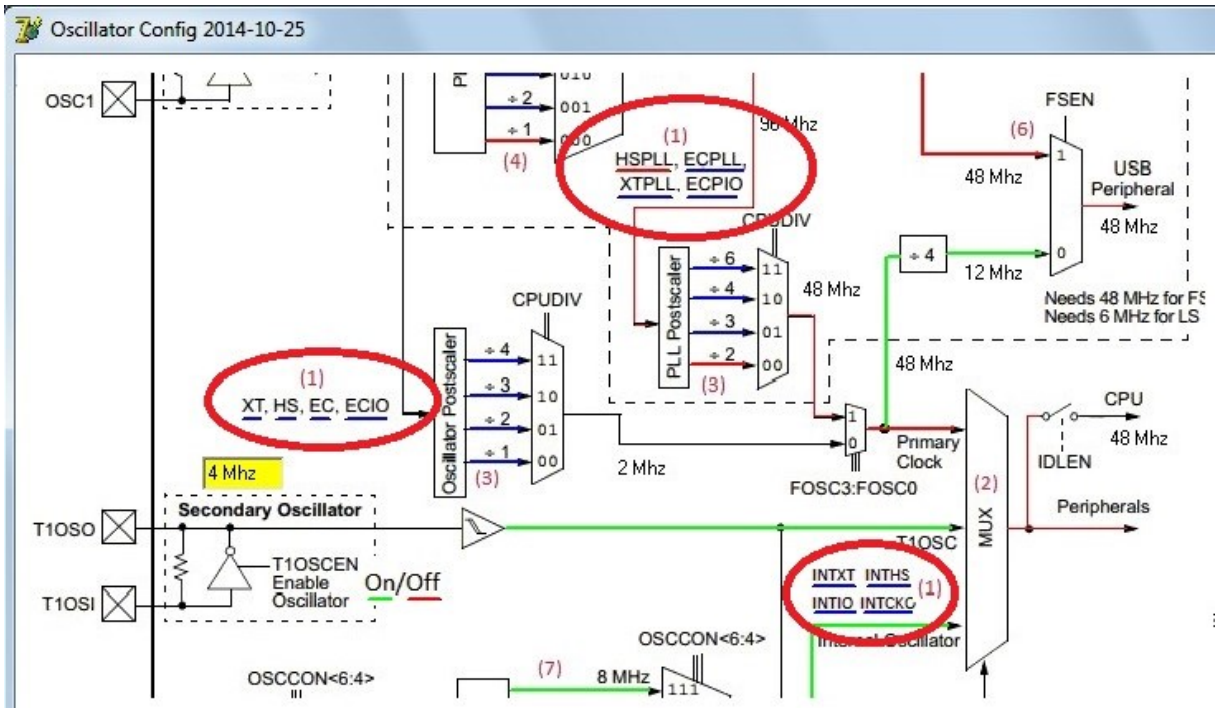
for some activities several different zones in the block diagram can be of importance, especially for action (1), the oscillator type selection.

In a number of cases (PIC’s) not all activities have to be done, because the PIC does not have the capability.

Please see the PIC’s datasheet to know about the names of oscillator types.

The main actions are:

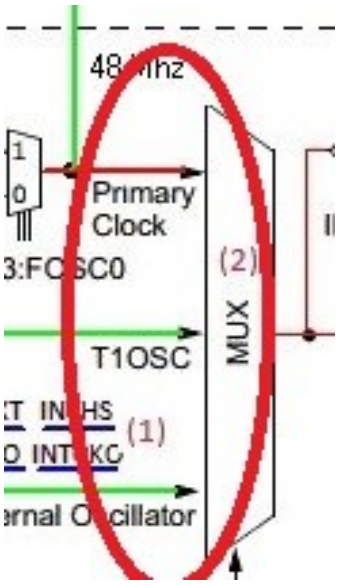
4.4.2 (1) Select the Oscillator type



Select the wanted Oscillator type from those available.

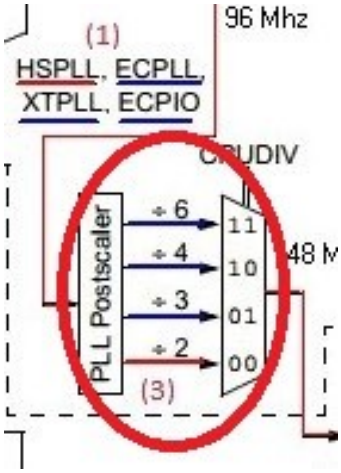
4.4.3 (2) Select the clock source

This is not always necessary, in some cases this is already done in the previous action.

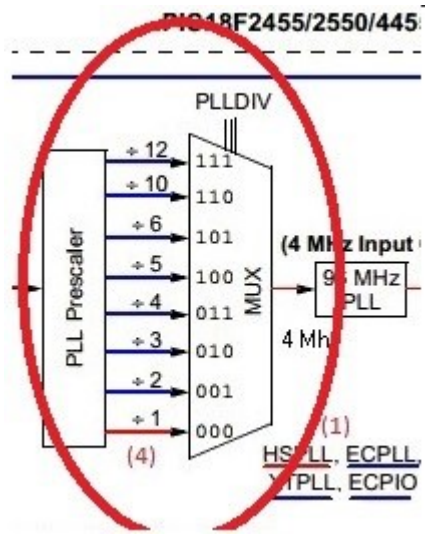


Select the MUX input to be used.

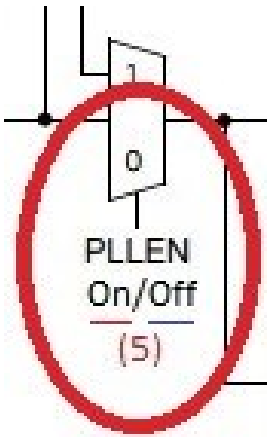
4.4.4 (3) Select the CPU clock divider



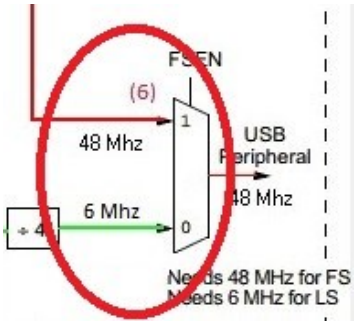
4.4.5 (4) Select the PLL prescaler (only with PLL osc type)



4.4.6 (5) Select the PLL Enable (only with PLL osc type)

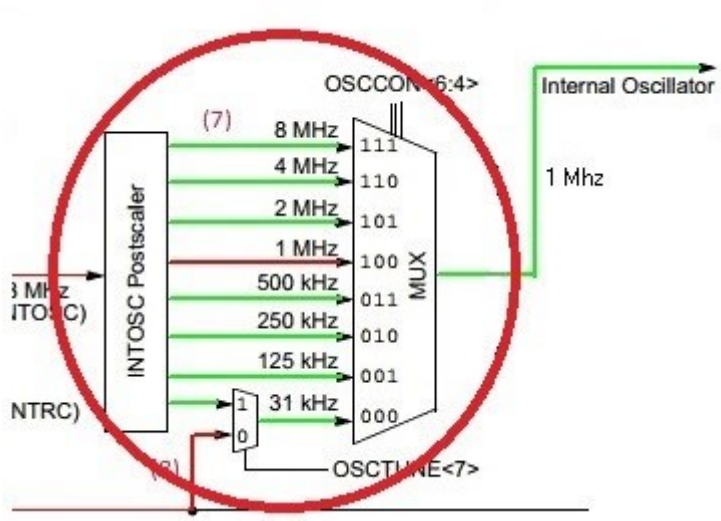


4.4.7 (6) Select Low speed USB clock (only types with USB)

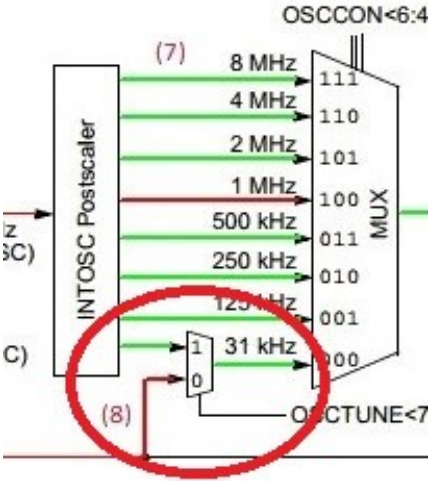


Choose Full speed USB (48 Mhz) or Low Speed USB (6 Mhz).

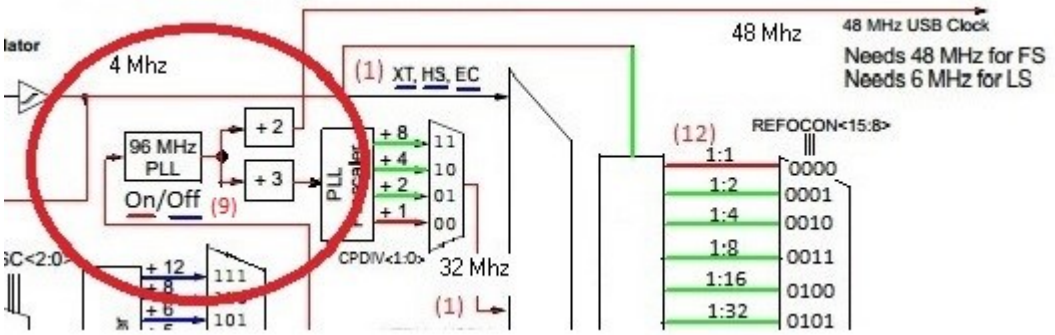
4.4.8 (7) Select the internal oscillator postscaler (only with int osc usage)



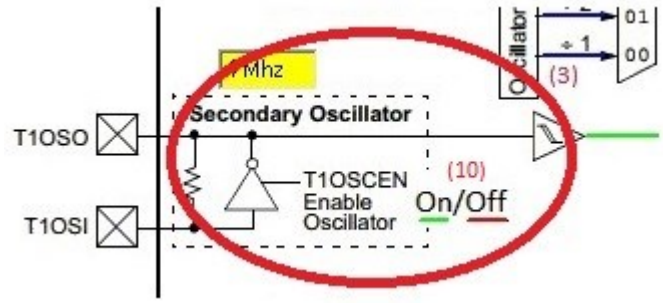
4.4.9 (8) Select the Internal Oscillator Low-Frequency Source Select bit (only with int osc usage)



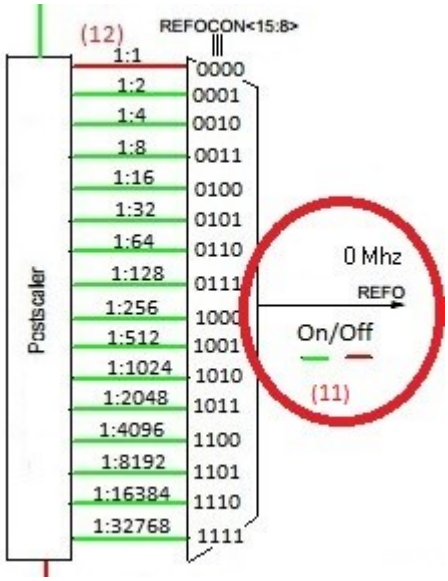
4.4.10 (9) Select the USB PLL yes/no (only with USB types)



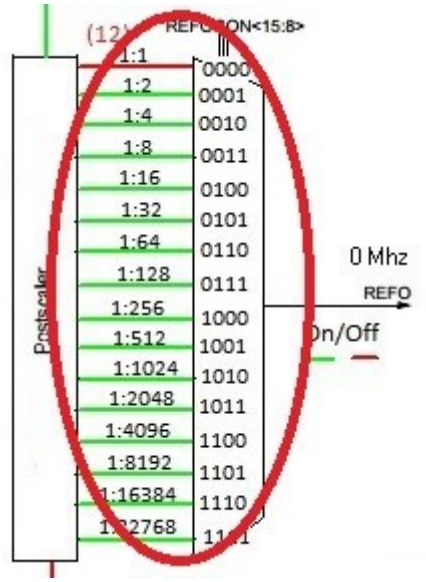
4.4.11 (10) Enable the oscillator(s) where needed



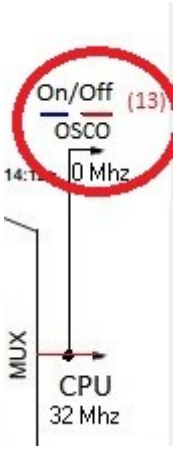
4.4.12 (11) Enable the Clock reference output



4.4.13 (12) Select the Clock reference divider

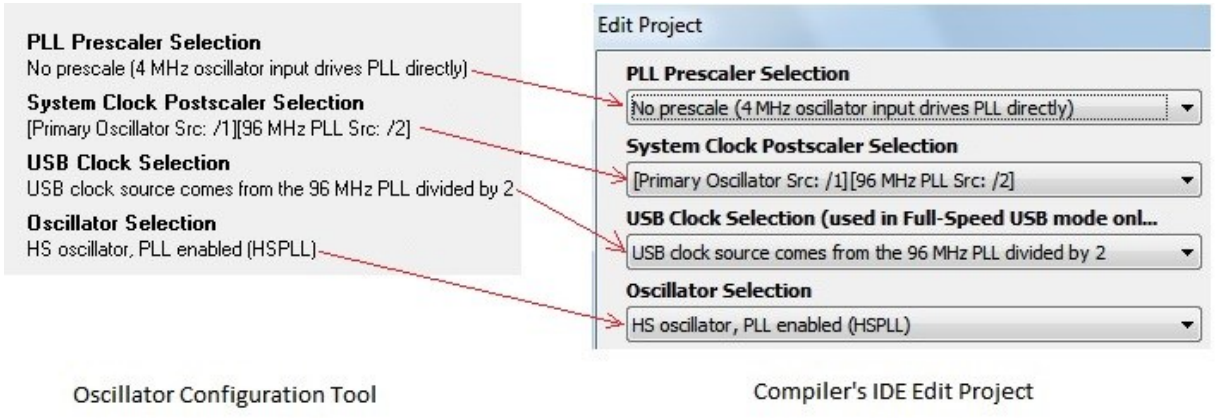


4.4.14 (13) Enable the Oscillator output



4.5 Copy the Configuration bits to the Compiler's IDE

Copy (=select) the Configuration bits outcome (visible at top right in the tool) in the Compiler's IDE "Edit Project " menu.



Oscillator Configuration Tool

Compiler's IDE Edit Project

4.6 Copy the register settings to your project's initialisation code

Copy the register settings (to be found in the "Code" menu, section 4.2.2) to your project's initialisation code (you can leave out the "default" code if you want) using the Clip Board functions.

Initialisation code

```
OSCCON := (OSCCON and (not 112)) or 64; // 1 Mhz (default)
OSCTUNE := (OSCTUNE and (not 128)); // 31 Khz from INTRC source (default)
OSCCON := (OSCCON and (not 3)); // Primary clock (default)
UCFG := (UCFG and (not 4)) or 4; // FSEN = 1
T1CON := (T1CON and (not 8)); // Disable Timer1 (secondary) oscillator (default)
```

4.7 Viewing clock frequencies along the clock path

If you want to see the clock frequencies along the clock path (especially the actual CPU clock, and/or the USB clock), then also fill in the frequency of the oscillator(s) in the input fields (yellow).

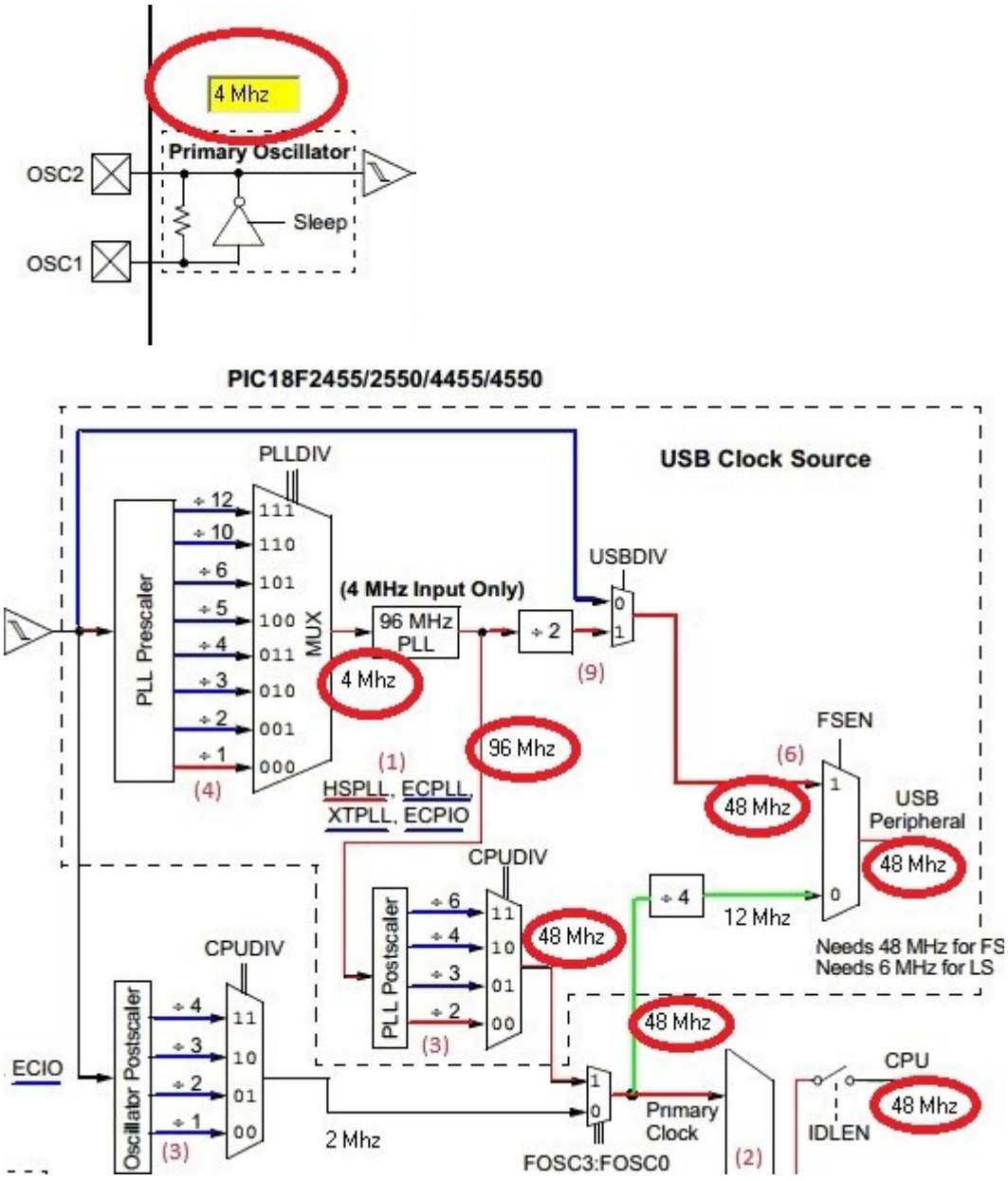


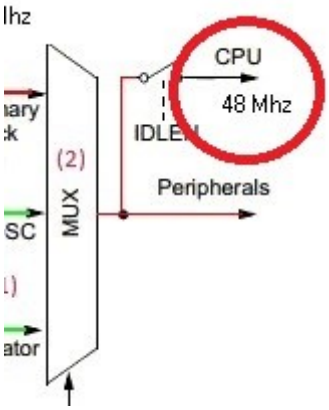
Figure 3

The tool shows the clock frequencies on several important points in the clock paths, see Figure 3. Mostly these clock frequencies are derived values, which can only be calculated correctly if the initial (crystal) frequencies are correctly filled in.

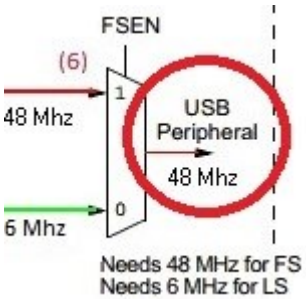
Remark: The decimal separator (for frequencies) is always a dot ('.'), regardless of the region you use the tool in.

The two most important clock frequencies are:

The "**CPU**" clock in the diagram: this is the frequency to be entered in the Compiler's IDE "Project Settings, **Frequency**", or "Edit Project, **MCU Clock Frequency**".



The USB clock. Only 2 frequencies are valid here: 6 Mhz for low speed USB and 48 Mhz for full speed USB. The tool does not check against these values, you will have to do it manually.



5 Adding PIC's

5.1 Overview

Adding a new PIC (family) can be done by:

- if the block diagram and the settings are equal to those of another already added family then the only thing to do is adding it to the file "Oscillator_Configuration_PICs.txt". One line should be added e.g.: `P18F25K22 = P18F45K22` where the left side is the name of the PIC to be added, and the right hand side is the PIC (family) of which the ..._lines.txt and ....bmp files are to be used.
- If no already existing .bmp and .txt file are describing the new PIC (family) then the following has to be done:
    - Create the oscillator block diagram (section 5.2)
    - Create the definitions text file (section 5.3)
    - Adapt the Pic's selection file "Oscillator_Configuration_PICs.txt" (section 5.4)

5.2 Creating the oscillator block diagram

This files is a copy (probably adapted) from the oscillator block diagram in the MicroChip datasheet. It is important to refer as much as possible to the datasheet concerning images/pictures and names used.

The file that should be created is a “Oscillator_<picname>.**bmp** file. Normally you would make this file by copying the block diagram from the datasheet.

 Keep in mind that everything that has to be “selectable” in the block diagram (e.g. a clock path or an on/off setting of a PLL) has to be represented by a “line” (segment). Normally this is already the case with clock paths (the block diagram in the datasheet is actually made to show them), bot for some other items the line segment has still to be added.

Make anyway sure that:

All oscillator types can be selected. If none or not all oscillator types can be selected (= are present in the diagram) then add them to the block diagram, e.g.:

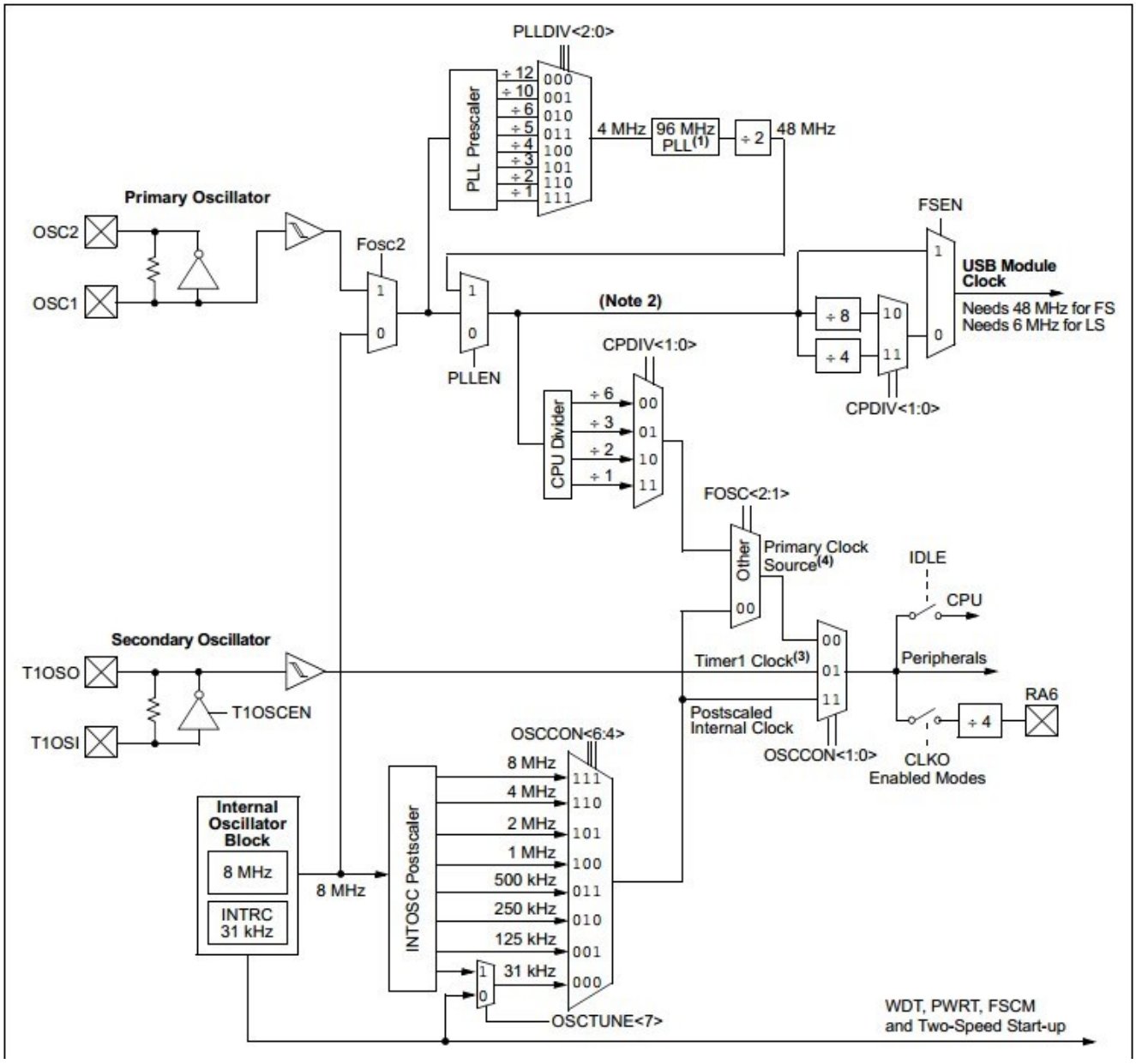
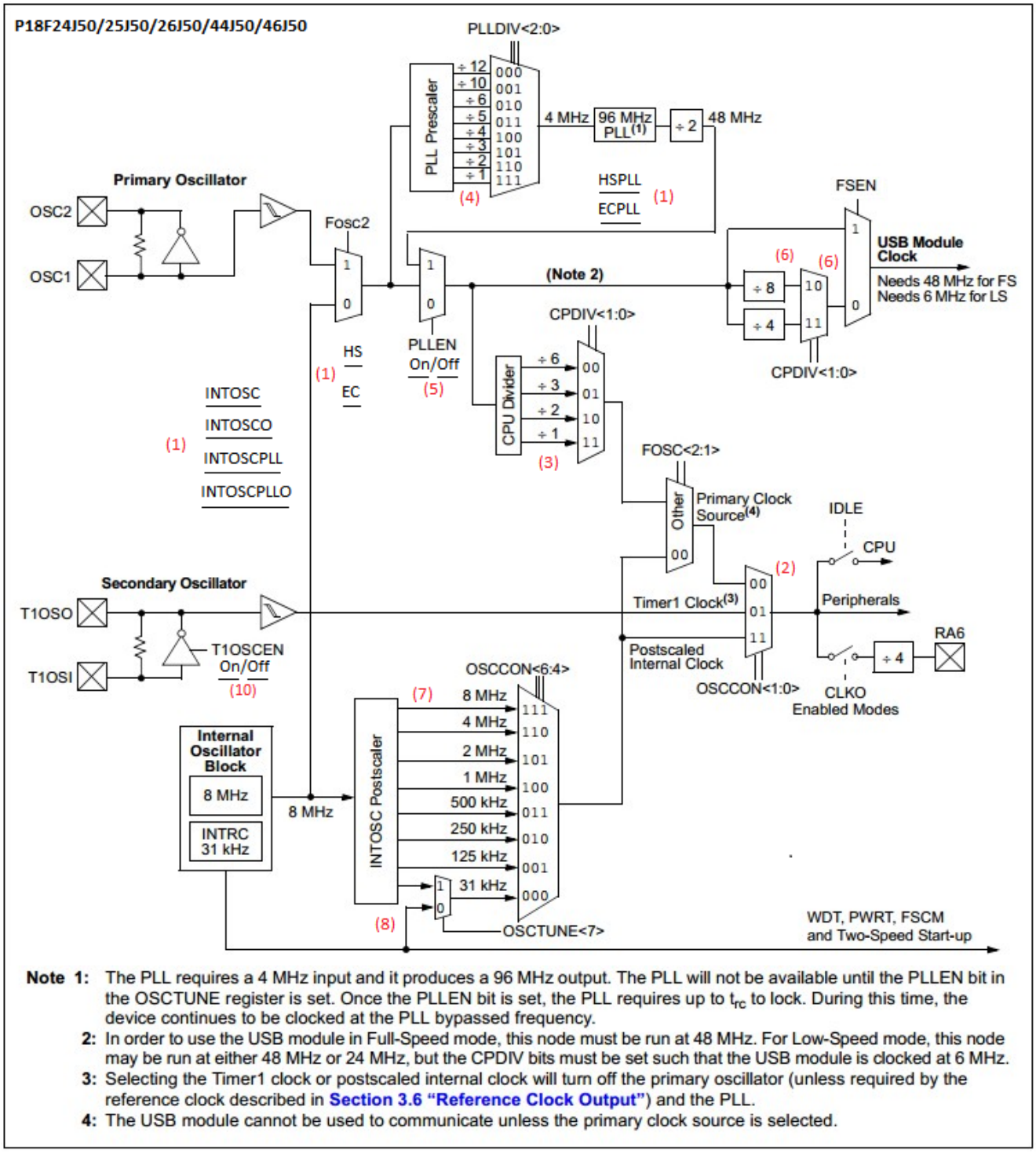


Figure 4 Without OscType selection

Above is how the oscillator diagram looks like in the datasheet of the PIC, the oscillator types are not mentioned.



Note 1: The PLL requires a 4 MHz input and it produces a 96 MHz output. The PLL will not be available until the PLLEN bit in the OSCTUNE register is set. Once the PLLEN bit is set, the PLL requires up to t_{rc} to lock. During this time, the device continues to be clocked at the PLL bypassed frequency.

2: In order to use the USB module in Full-Speed mode, this node must be run at 48 MHz. For Low-Speed mode, this node may be run at either 48 MHz or 24 MHz, but the CPDIV bits must be set such that the USB module is clocked at 6 MHz.

3: Selecting the Timer1 clock or postscaled internal clock will turn off the primary oscillator (unless required by the reference clock described in **Section 3.6 "Reference Clock Output"**) and the PLL.

4: The USB module cannot be used to communicate unless the primary clock source is selected.

Figure 5 With OscType selections

See (1) in the above diagram where the selection possibility of the oscillator types have been added. See the datasheet for the oscillator types available in the PIC. It is best to add an "underscore line" to each oscillator type so the user will know where to click to select a type.

All selections that together define the oscillator settings can be made in the diagram. Sometimes the datasheet shows a “simplified” block diagram, which should be extended to make all selections possible, e.g.:

FIGURE 8-1: PIC24FJ64GA004 FAMILY CLOCK DIAGRAM

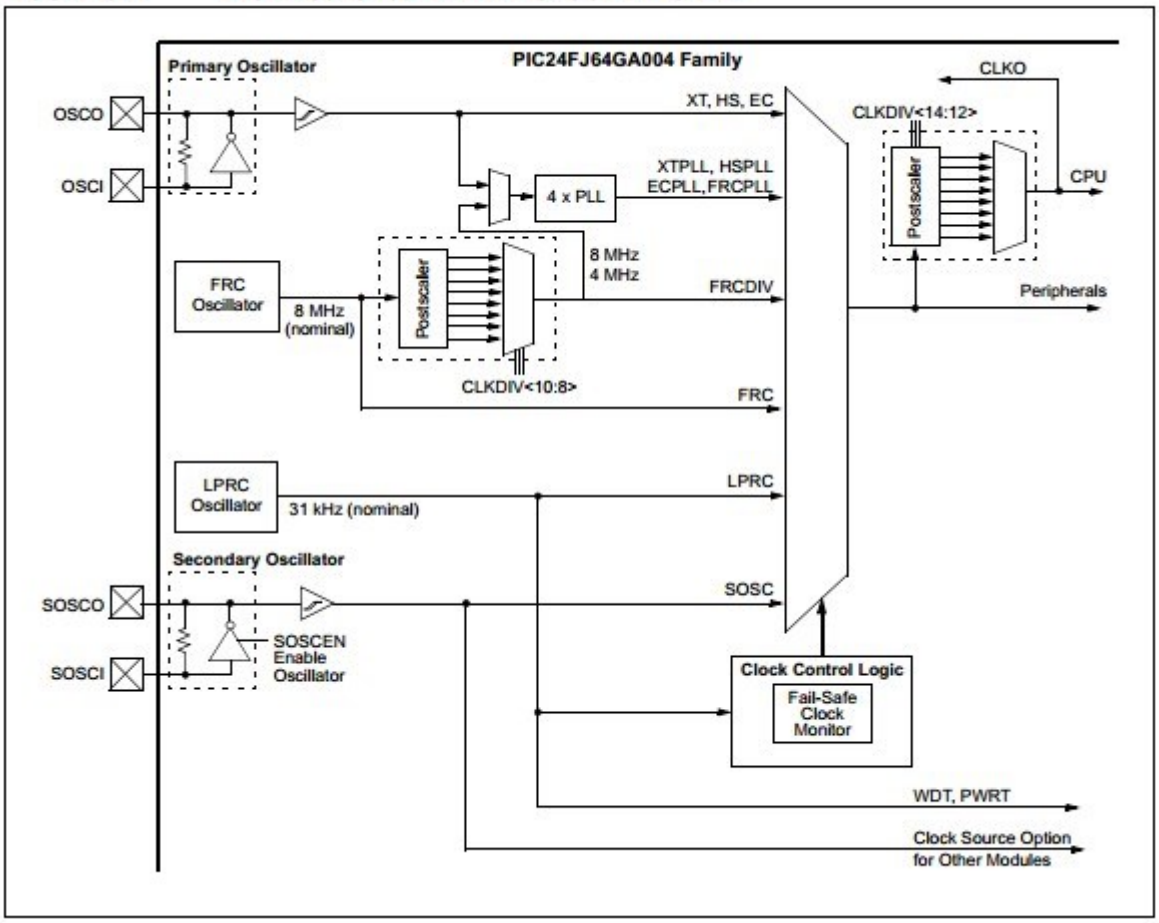


Figure 6 Simplified blockdiagram

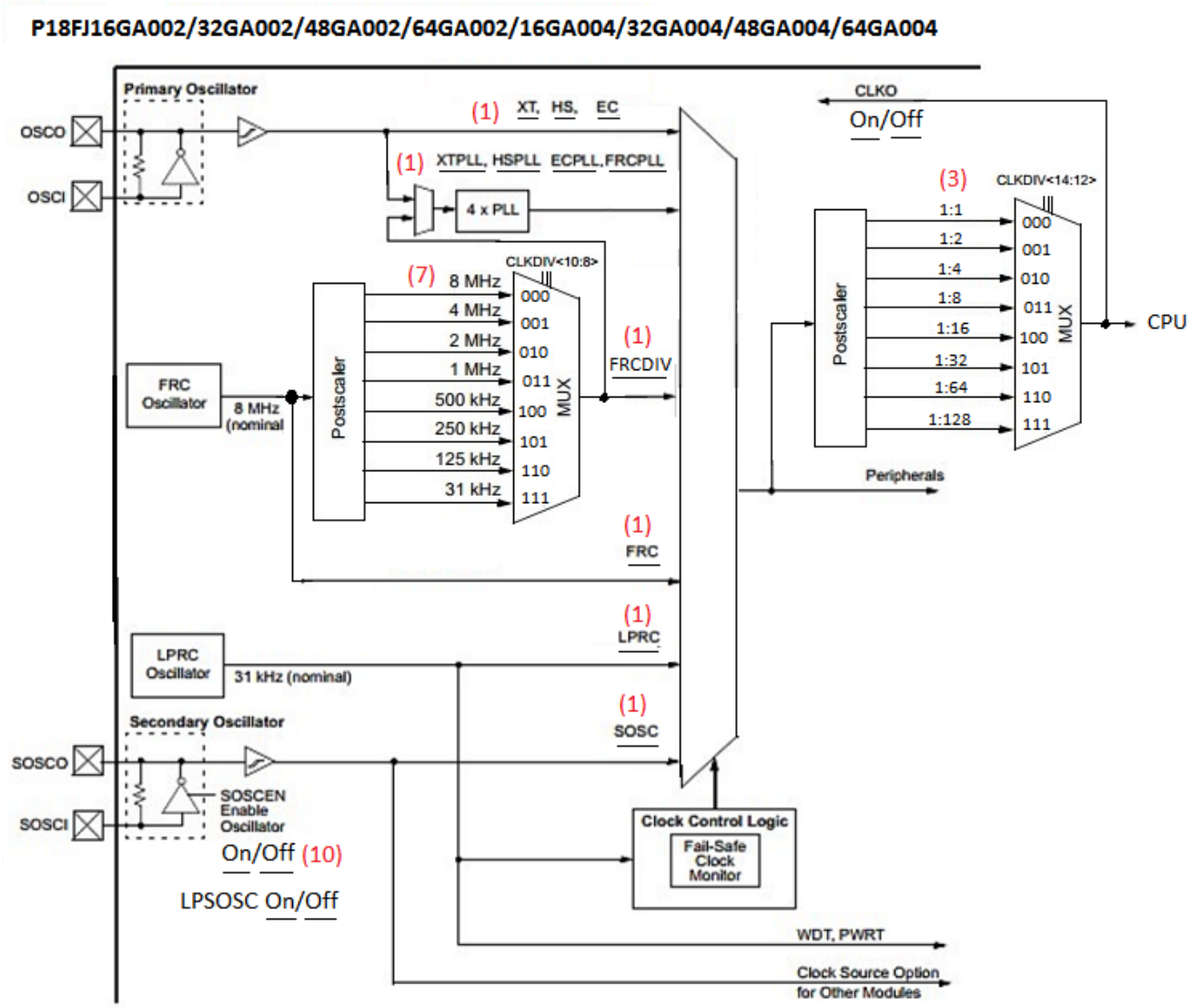


Figure 7 Extended block diagram

See (3) and (7) in the above diagram for simplifications that have been expanded.

Settings/selections that are not available in the block diagram are added, e.g. PLL On/Off, Oscillator(s) On/Off: see (10) and "CLKO On/Off" in Figure 7 Extended block diagram.

Not obligatory: add a number to the diagram –e.g. (2) – where needed to make clear what type of selection can be made. The numbers used should be the same as in the tool's help file (and section 4.4.2 and following).

5.3 Creating the definitions text file.

This file, needed for every PIC family supported, is named "**<picname>_lines.txt"**.

This is the most difficult part of the process, it consists of

1. Defining selectable lines for configuration bits settings, (see section 5.3.1)
2. Defining selectable lines for register settings (used for code initialisation), (see section 5.3.2)
3. Defining which "groups" of lines are mutually exclusive, (see section 5.3.3)
4. Defining lines that "follow" lines of the above 2 classes, this type of line is not selectable, (see section 5.3.4)
5. Formulas used for showing clock frequencies along clock paths. (see section 5.3.5)

Definitions 1. and 2. can be stated in the file in an arbitrary order (even interleaved). The rest of the definitions has to be stated in the sequence as shown above.

All definition types stated above can be left out if not needed/not applicable.


Important remark: For definitions 1. and 2. in the table above: the first time a definition appears in the file (either for a configuration or a register settings) defines which value will be taken after selecting a PIC to show in the oscillator block diagram.

5.3.1 Lines for Configuration bits

Lines for configuration bits are shown in the block diagram as thick blue lines (when unselected). If they are selected they show as thick red lines.

Each line has the following format:

<name>, <group>, <id>, <X1>, <Y1>, <X2>, <Y2>, <excl>, <incl>, <value>[, <formulavalue>]

As you can see the latter value is optional. All items in a line are separated by comma's.

The fields are:

- <**name**>: The name of the configuration setting. This is usually a part of a configuration word. The <name> must be the same as the one defined in the processors **.mlk** file, e.g.: PLLDIV, CPUDIV, FOSC etc... So, do NOT use the names as they are shown in the oscillator block diagram.
  Important: the <name> should not define a bit field (e.g. <5:3>), the bit field within the config setting is fetched from the PIC's .MLK file.
- <**group**>: This value denotes a group number: the same number should be used for all lines in the same group. Give all lines closely connected in behavior the same group number, e.g. give all lines with name PLLDIV group number 0 (you can start with 1 also). It is advisable to give the lines which define the oscillator type a different group number.
  See sections 5.3.3 and 5.3.4 for the usage of <group>.

- <**id**>: not used yet.
- <**X1**><**Y1**><**X2**><**Y2**>: The coordinates of the line on the oscillator's block diagram image , see section 5.5
- <**excl**>: a number (a kind of group number) of mutual exclusive lines. If you give 2 lines (no matter to which <group> they belong) the same number, then selecting one of them will unselect the other one. Used where selections are mutually exclusive (e.g. PLLDV or the Oscillator type).
  If exclusive lines are within one group then it is easier (better understandable) if the number equals the <group> of the line, but not obligatory.
  Using -1 means the line is not mutual exclusive with any other line.
  Do NOT make lines both mutual exclusive and inclusive!
- <**incl**>: a number (also a kind of group number) of inclusive lines. Inclusive lines appear always selected together or unselected together. Used where a number of lines can be used to select the same path.
  If inclusive lines are within one group then it is easier (better understandable) if the number equals the <group> of the line, but not obligatory.
  Using -1 means the line does not become also selected/unselected when another line becomes selected/unselected.
  Do NOT make lines both mutual exclusive and inclusive!
- <**value**>: This is the value that will be selected as config bits content.
  Important: The value defined here should be the same as defined in the .mlk file [1] for a certain configuration item.
- <**formulavalue**> (optional): This value is only used in the clock frequency formulas. It is the "logical" counterpart of the previous value. E.g. <value> can be $00 or $10 (PLLCFG of the P18F45K22), meaning "PLLCFG = 0" or "PLLCFG = 1" in the configuration bits, but the clock frequency calculations need a actually value of 0 or 1 to do calculations, not the $00 or the $20 that are used as the config bits.
  If the <formulavalue> is omitted then <value> is taken to do the oscillator frequency calculations.

Example (P18F45K22):

```
PRICLKEN,  2,   0,  48, 157,  57, 157,   2,    -1,   $20, 1 // PRICLKEN = 1; primary clock always enabled
PRICLKEN,  2,   0,  65, 157,  78, 157,   2,    -1,   $00, 0 // PRICLKEN = 0; primary clock controlled by PRISD
```

Both lines defined above affect the configuration setting "PRICLKEN", the top line will set the value $20 into the configuration word containing "PRICLKEN", the lower one will set the value $00 to it. Both values can be found in the .mlk file.

The values used in the clock frequency calculations are resp. 1 and 0.

Both lines belong to the same <group> 2, and they are mutually exclusive (they have both <excl> set to 2).

 See also this Important Remark when you want to set more than one value selecting a single line.

[1] So, the value is depending on the bit field position in the config word.

5.3.2 Lines for register settings

Lines for configuration bits are shown in the block diagram as thick green lines (when unselected). If they are selected they show as thick red lines.

Each line has the following format:

<name>, <group>, <id>, <X1>, <Y1>, <X2>, <Y2>, <excl>, <incl>, <value>[, <formulavalue>]

As you can see the latter value is optional. All items in a line are separated by comma's.

The fields are:

- <**name**>: The name of the register setting. This name should contain a valid register name for the PIC concerned. The correct name can be found in the **PIC's .mpas** file in the defs directory, do not use the name mentioned in the oscillator's block diagram.
  Additionally the name must have a suffix indicating the bit field inside the register, e.g. <5:3>, meaning the bit field ranges from bit 5 to bit 2 (so, 3 bits wide). If the field is only 1 bit wide then also the suffix e.g. <4> can be used (here indicating bit 4). Important: the <value> to give here is the value to be placed in the bit field, not the value to be placed in the whole register.
- <**group**>: See section 5.3.1
- <**id**>: not used yet.
- <**X1**><**Y1**><**X2**><**Y2**>: The coordinates of the line on the oscillator's block diagram image, see section 5.5
- <**excl**>: See section 5.3.1
- <**incl**>: See section 5.3.1
- <**value**>: This is the value that will be put into the bit field of the register by the initialisation code (to be seen with the "Code" menu button). [2]
- <**formulavalue**> (optional): See section 5.3.1

Example (P18F45K22):

```
OSCCON<6:4>, 8,   0,  117, 417, 254, 416,   8,   -1,   3,   1 // IRFC = 3; INTOSC HF 1 Mhz (default)
OSCCON<6:4>, 8,   0,  117, 379, 254, 379,   8,   -1,   7,  16 // IRFC = 7; INTOSC HF 16 Mhz
OSCCON<6:4>, 8,   0,  117, 389, 254, 388,   8,   -1,   6,   8 // IRFC = 6; INTOSC HF 8 Mhz
OSCCON<6:4>, 8,   0,  116, 398, 254, 399,   8,   -1,   5,   4 // IRFC = 5; INTOSC HF 4 Mhz
OSCCON<6:4>, 8,   0,  117, 407, 254, 407,   8,   -1,   4,   2 // IRFC = 4; INTOSC HF 2 Mhz
OSCCON<6:4>, 8,   0,  215, 457, 254, 456,   8,   -1,   2, 0.5 // IRFC = 2; INTOSC 500 Khz
OSCCON<6:4>, 8,   0,  214, 499, 254, 499,   8,   -1,   1, 0.25 // IRFC = 1; INTOSC 250 Khz
OSCCON<6:4>, 8,   0,  186, 550, 254, 550,   8,   -1,   0, 0.03125 // IRFC = 0; INTOSC 31.25 Khz
```

[2] So, the value should not take into account the bit field shift (which is already defined in the <name>).

All lines defined above affect the register setting of bits 6..4 in register “OSCCON” (which is the IRFC field).
All lines belong to the same <group> 8, and they are all mutually exclusive (they all have <excl> set to 8).
Here one can see clearly the difference between <value> and <formulavalue>, the latter one being the “meaning” of the register<value>.
Comments added in register lines are copied to the generated initialisation code.


Important Remark: It is possible to set more than one value (config bits or/and register settings) when selecting “one line”. The method to do this is placing 2 or more lines on top of each other, e.g.:

```
OSCTUNE<7>, 13,    0,  117, 561, 171, 561,   11,   -1,    0 // INTSRC  = 0 (default)
OSCCON2<4>, 13,    0,  117, 561, 171, 561,   12,   -1,    0 // MFIOSEL = 0 (default)
```

As you can see the coordinates of both lines are the same, so both values will be set after the lines become selected. Configuration bit lines and register setting lines can also be placed on top of each other. Do never make lines on top of each other mutual exclusive with each other.

5.3.3 Mutual exclusive groups

The possibility exists to define mutually exclusive “groups”. Here “mutually exclusive” means:
when a line in a group becomes selected all lines in the mutually exclusive groups become deselected.

The format of such line is

“EXCL”, <group>, <group>[, <group>[, <group>[,...]]]

Example:

```
EXCL, 5, 6, 7
```

Above line means: the lines of groups 5, 6 and 7 are mutually exclusive.

5.3.4 Follow lines

This functionality does not generate selectable lines on the block diagram. It simply allows certain lines to be marked in red according lines that have already been selected, hence the term “following”. This type of line is shown in small black or small red, and is meant to “complete” the selected path in the block diagram.

The line is shown in read when at least one line in one of the stated groups is selected.

The format of the line is:

“FOLLOW”, <X1>, <Y1>, <X2>, <Y2>, <group>[<group>[<group>[...]]]
Important: the groups themselves are separated by spaces!

Example:

```
FOLLOW, 464, 231, 511, 231, 5 6 7
```

The line on coordinates 464, 231, 511, 231 follows the groups 5, 6 and 7. If shows red when one or more lines in one or more of the three groups is/are selected.

There is also an “inverse” of this function:

“FOLLOW_NOT”, <X1>, <Y1>, <X2>, <Y2>, <group>[<group>[<group>[...]]]

meaning: the line shows red when none of the lines in the stated groups is selected.

Example:

FOLLOW_NOT, 292, 247, 351, 247, 3 12 // PLL disabled

The line on coordinates 292, 247, 351, 247 becomes red if none of the lines in groups 3 and 12 are selected.

5.3.5 Clock frequency formulas

These formulas are evaluated in the same order as they are defined in the definitions text file. The calculation occurs when a line becomes activated.

The format of a “formula” line is the following:

“SHOWVAL”, <X>, <Y>, <Ex>[“=” <formula>]

items are:

- <X>, <Y>: the coordinates on the block diagram where the (calculated) values are to be shown (see also section 5.5.
- <Ex>: an “evaluation number” from E0..E99. These values will be shown in the block diagram.
- <formula>: a mathematic expression wherein config items, register items and <Ex> values can be used. Additionally a number of functions is available.
  The effect of evaluating the line is that <Ex> gets the value of the formula.
  If the formula is omitted, the value is shown as an editable input field. This should be e.g. the case for the oscillator frequency. Input fields are shown with a yellow background.

Examples:

SHOWVAL, 323, 81, E2 = OSCCON2<3> * E1 Here E2 gets the value selected for OSCCON2<3> times the value of E1 that was calculated in some previous formula line.

SHOWVAL, 57, 209, E0 Here there is no formula, this means the field is presented as an input edit field to be filled by the user with the actual value.

5.3.5.1 Operators in formulas

In the formula part a number of operators (which always have only one character) are available:

'+', '-', '*', '/', '^', '%', '=', '>', '<', '|', '&'

All operators are binary operators: they have 2 operands, e.g. “a + b” where a and b are the operands.

Most of the operators are supposed known, but a few are special:

^: the result of “a ^b” is “the value of a raised to the power b”, e.g. “10 ^ 2” would be 100.

%: the result of “a % b” is “(a/b)*100” (“percent” value), so “200 % 1000” gives 20.

| : logical “or”: the operands are boolean (or boolean expressions), the result is boolean

&: logical “and”: the operands are boolean (or boolean expressions), the result is boolean

5.3.5.2 Functions in formulas

The evaluation module has a number of functions built in which can be used in the formulas. The general format of a function is:

<functionname>(list of arguments separated by commas).

A function mostly returns a real (float) value.

Until now only 2 types of functions have been used in the formulas:

- "Ifthen"(arg0, arg1, arg2) "IfThen" checks the expression passed as arg0 and returns arg1 if it evaluates to +1 or more, or arg2 if it evaluates to 0. Arg0, arg1 and arg1 can in turn be complex expressions.
  Example: ifthen((OSCCON<8:5> > 15), E10, E11) means: returns E10 if OSCCON<8:5> exceeds 15, else return E11 as the result of the function. Note the parenthesis around the first argument.
- "select"(arg0, arg1, ...) "Select" returns arg1 if arg0 is 0, arg2 is arg0 is 1 etc.
  Example: select(OSCCON<1:0>, E7, E2, E4) means: return E7 if the value of OSCCON<1:0> is 0, E2 if it is 1 and E4 if it is 2. All arguments can be in turn complex expressions.
  Make sure there are arguments enough for all possible values of arg0!

For other available functions please inspect the Delphi code of the "RcsEval" unit.


Important note: Each formula line is evaluated from left to right, there is no precedence. So: ALWAYS use parenthesis in "complex" expressions. So, "a +b > c + d" should be "(a+b) > (c+d)", otherwise it will be evaluated as "((a+b) > c) + d" which is probably not wanted.

5.4 Adapting the PIC's selection text file

This is the easiest part of the process. The content of this file defines the content of the "PIC" menu.

The only thing to do here is adding the new PIC (family) to the file "Oscillator_Configuration_PICs.txt". One line should be added e.g.: `P18F25K22 = P18F45K22` where the left hand side is the name of the PIC to be added, and the right hand side is the PIC (family) of which the ..._lines.txt and ....bmp files are to be used.

Since one .bmp file and one .txt file combination will describe a number of individual Pic's usually a number of lines have to be added referring to the .bmp and .txt files created. E.g.:

```
P18F65J50 = P18F26J50
P18F66J50 = P18F26J50
P18F66J55 = P18F26J50
P18F67J50 = P18F26J50
P18F85J50 = P18F26J50
P18F86J50 = P18F26J50
P18F86J66 = P18F26J50
P18F87J50 = P18F26J50
```

In above part of the file 8 PIC's in total use the files "Oscillator_P18F26J50.bmp" and "Oscillator_P18F26J50_lines.txt".

A single '-' (minus sign) in the list in "Oscillator_Configuration_PICs.txt" displays an empty line in the PIC selection menu, e.g.:

```
P18F47J53 = P18F27J53
-
P18F65J50 = P18F26J50
```

5.5 Addendum: line coordinates

The coordinates of lines and Ex values are needed in the definitions in the text file.

There are 2 methods to achieve this coordinates:

- In the bottom right corner of the tool's main screen the current coordinates of the cursor (provided it is inside the oscillator block diagram) are displayed. By moving the cursor to the start of a line, to the end of a line or to the position of a Ex value, the coordinates can be obtained.
- Especially for lines: fetch the coordinates of a line into the Clipboard, ready to place inside the text file, doing the following:
    - Place the cursor on the top/left of the line and press the mouse's scroll wheel,
    - Place the cursor on the bottom/right of the line and press the mouse's right hand key,
    - Goto the place in the text file where the coordinates are to be inserted and press ctrl-V.

[end of document]