

```

#define T_SGR_MIN    (500e-9)           // full precision
#define T_SGR_MIN_R  (499.188899e-9)    // reduced precision, calculated with float/hex Tool by myself

#define I_MIN        (0.1e-3)           // full precision
#define I_MIN_R      (99.658966064453125e-6) // reduced precision, calculated with float/hex Tool by myself

```

I have defined two constants in my code with full and reduced precision. The values with the reduced precision should be calculated later with the preprocessor. The values with the reduced precision yields to a more efficient code, because the lower 16bit of the floating point value are zero and the compiler is able to use the mnemonics with the integrated #16FHi constant.

Code with the full precision constants

The screenshot shows the Code Composer Studio interface with the following details:

- Project Explorer:** Shows the file `sqrt.cla` and the assembly file `sqrt.cpu01c`.
- Registers:** Shows the state of core registers for CPU1_CLA1, including FPR0, FPR1, and FPR2.
- Memory Map:** Shows the memory map for the application.
- Console:** Shows the output of the code execution, including GEL output and memory map initialization.

The screenshot shows the Code Composer Studio interface with the following details:

- Project Explorer:** Shows the file `sqrt.cla` and the assembly file `sqrt.cpu01c`.
- Registers:** Shows the state of core registers for CPU1_CLA1, including FPR0, FPR1, and FPR2.
- Memory Map:** Shows the memory map for the application.
- Console:** Shows the output of the code execution, including GEL output and memory map initialization.

Code with the reduced precision constants

The screenshot shows the Code Composer Studio interface with the following panes:

- Project Explorer**: Shows the project structure with files like `cla_sqrt_cpu01.c` and `cla_sqrt_cpu01.s`.
- Disassembly**: Displays assembly code for the `cla_sqrt_cpu01` project. The assembly code includes instructions like `arg = _mmmaxf32(t_p - (T_F_MAX-T_TM_CORR) , 0.0);` and `arg = _mmmaxf32(t_p - (T_F_MAX-T_TM_CORR) , 0.0);`. A red arrow points to the instruction `arg = _mmmaxf32(t_min_r);`.
- Registers**: Shows the state of CPU registers. A red arrow points to the `M00V12` register, which contains `0x03d81d`.
- Memory Dump**: Shows memory dump data for the `CLA1` core, including memory addresses like `784036c4` and values like `0x03d81d`.
- Console**: Displays GEL output and memory initialization status.
- Debug**: Shows the debug session for the `cla_sqrt_cpu01` project.
- Breakpoints**: Shows the current breakpoints in the code.
- Problems**: Shows any build or runtime errors.

With the reduced precision only one mnemonic is needed to perform the operation.