Six of the 32 extra working registers can be used to form a combination of three 16-bit registers. These 16-bit registers can be used to access external memory and FLASH programs by indirectly addressing the address pointer. The LGT8XM supports single-cycle 16-bit arithmetic, greatly improving the efficiency of indirect addressing. The three special 16-bit registers in the LGT8XM core are named X,Y, Z registers, which are described in more detail later.

The ALU supports arithmetic logic operations between registers and between a constant and a register. The operation of a single register can also be performed in the ALU. After the ALU operation is completed, the effect of the operation on the state of the system is updated to the status register (SREG).

Program flow is provided by conditional and unconditional jump and call instructions, able to directly address the whole address space. Most LGT8XM instructions are 16 bits. Each program address space corresponds to a 16-bit or 32-bit LGT8XM instruction.

After the system responds to an interrupt or a subroutine call, the return address Program Counter (PC) is stored on the stack. The Stack is effectively allocated in the general data SRAM, and consequently the Stack size is only limited by the total SRAM size and the usage of the SRAM. All applications that support interrupt or subroutine calls must first initialize the Stack Pointer Register (SP), which can be accessed through the IO space. Data SRAM can be accessed in five different addressing modes.

The internal memory of the LGT8XM is linearly mapped to a uniform address space. Please refer to the the storage memory chapter for further details.

The LGT8XM includes a flexible interrupt controller. It can be controlled by a global interrupt enable bit in the Status Register. Every interrupt has a separate interrupt vector. The priority of the interrupt has a corresponding relationship with the interrupt vector address. The smaller the interrupt address, the higher the priority of the interrupt.

The I/O space contains 64 register spaces that can be directly addressed by IN/OUT instructions. These registers are used for kernel control as well as control functions for status registers, SPI and other I/O peripherals. This space can be accessed directly or by their address mapped to the data memory space (0x20 - 0x5F). In addition, the LGT8FX8P also includes extended I/O space, which is mapped to data memory space 0x60-0xFF, which can only be accessed using ST/STS/STD and LD/LDS/LDD instructions.

To enhance the computing power of the LGT8XM core, a 16-bit LD/ST extension has been added to the instruction set. This 16-bit LD/ST expansion works with the 16/32-bit Dimensional Operation Acceleration Unit (uDSU) For efficient 16-bit data operations. At the same time, the kernel also increases the 16-bit access capability to the RAM space. The 16-bit LD/ST extension can pass 16 bits of data between the uDSU, RAM, and working registers. Please refer to the "Digital Operation Accelerator" section for details.

ALU - Arithmetic Logic Unit

The LGT8XM internally contains a 16-bit arithmetic logic unit that can perform 16 bit arithmetic operations on data in one cycle. The highly efficient ALU is connected to 32 general purpose working registers. The ALU has the ability to perform two arithmetic operations between registers or registers and immediate data in one cycle. There are three types of ALU operations: arithmetic, logic, and bit operations. The ALU also includes a single-cycle hardware multiplier that implements direct signed or unsigned operations on two 8-bit registers in a single cycle. Please refer to the detailed description in the instruction set section.

SREG - System Status Register

The status register mainly stores the result information generated by the execution of the most recent ALU operation. This information is used to control the program execution flow. The status register is updated after the ALU operation has completely ended. This usually eliminates the need for separate compare instructions, resulting in a more compact and efficient code implementation. The value of the status register is not automatically saved and restored in response to an interrupt and exit from the interrupt, which requires software to implement.