RISC-V Assembler Documentation

1 How to Use

To create an executable named `riscv_asm`, follow these steps:

1. **Compile the Files:** Use the command below to compile all files that have been modified:

   ```
   make all
   ```

   This compiles the files that have been changed and then links them all together.

2. **Compile Specific Files:** To compile a specific target, use:

   ```
   make k.o
   ```

   This compiles only if that respective target's dependencies have been changed.

3. **Run the Executable:** Execute the file by running:

   ```
   ./riscv_asm
   ```

4. **Clean Up:** Remove all object files and the main executable with:

   ```
   make clear
   ```

2 File Structure

- `main.cpp`: Takes input from `input.s` and converts the instructions to their respective hex code by calling a function in `sort_find.cpp`. The output is stored in `output.hex`.
- `sort_find.cpp` & `sort_find.hh`: Processes input lines and directs them to corresponding type functions (R, S, J, I, U, B).
- `commonalgo.cpp` & `commonalgo.hh`: Contains functions for various conversions:
  - Convert decimal integer to binary string.
  - Convert register to binary format required for hex.
  - Convert binary string to a decimal number.
  - Convert decimal number to hexadecimal string.
  - Search for label line numbers.

3 Instruction Conversion Files

Each file handles a specific type of instruction conversion:

- `Rconvert.cpp` & `Rconvert.hh`: Performs operations for R-type instructions.

```
std::string Rconvert(std::string s, int line_counter);
```

- `Iconvert.cpp` & `Iconvert.hh`: Handles I-type instructions.

```
std::string Iconvert(std::string s, int line_counter);
```

- `Sconvert.cpp` & `Sconvert.hh`: Handles S-type instructions.

```
std::string Sconvert(std::string s, int line_counter);
```

- `Bconvert.cpp` & `Bconvert.hh`: Manages B-type instructions.

```
std::string Bconvert(std::string s, int line_number, struct store_label label_line[]);
```

- `Jconvert.cpp` & `Jconvert.hh`: Manages J-type instructions.

```
std::string Jconvert(std::string s, int line_counter, struct store_label label_line[])
;
```

- `Uconvert.cpp` & `Uconvert.hh`: Handles U-type instructions.

```
std::string Uconvert(std::string s, int line_counter);
```

- `Struct.hh`: Defines a global struct for storing line numbers and label names.

```
#ifndef struct_h
#define struct_h
#define MAX 1000
struct store_label
{
   std::string label;
   int label_line_num;
};
#endif
```

4 Common Functions

The following functions are extensively used across different files to assist in the conversion process:

- `std::string deci_to_bi(int x, int no_of_bits);`
- `std::string register_to_bi(std::string s, int line_counter);`
- `unsigned int binary_to_decimal(std::string s);`
- `std::string decimal_to_hex(unsigned int decimal);`
- `int search_label(std::string label, struct store_label label_line[]);`