**API Documentation**

**SRAM**

**Function:**

* void **init**(int size);
* uint16\_t **getSize**();
* uint8\_t **\*array**(int address);
* uint8\_t **readByte**(int address);
* bool **writeByte**(int address, uint8\_t value);
* int8\_t **readChar**(int address);
* bool **writeChar**(int address, int8\_t value);
* uint8\_t **readUChar**(int address);
* bool **writeUChar**(int address, uint8\_t value);
* int16\_t **readShort**(int address);
* bool **writeShort**(int address, int16\_t value);
* uint16\_t **readUShort**(int address);
* bool **writeUShort**(int address, uint16\_t value);
* int32\_t **readInt**(int address);
* bool **writeInt**(int address, int32\_t value);
* uint32\_t **readUInt**(int address);
* bool **writeUInt**(int address, uint32\_t value);
* int32\_t **readLong**(int address);
* bool **writeLong**(int address, int32\_t value);
* uint32\_t **readULong**(int address);
* bool **writeULong**(int address, uint32\_t value);
* int64\_t **readLong64**(int address);
* bool **writeLong64**(int address, int64\_t value);
* uint64\_t **readULong64**(int address);
* bool **writeULong64**(int address, uint64\_t value);
* float **readFloat**(int address);
* bool **writeFloat**(int address, float value);
* double **readDouble**(int address);
* bool **writeDouble**(int address, double value);
* bool **readBool**(int address);
* bool **writeBool**(int address, bool value);
* size\_t **readString**(int address, char \*value, size\_t maxLen);
* bool **writeString**(int address, const char \*value);
* String **readString**(int address);
* bool **writeString**(int address, String value);
* size\_t **readBytes**(int address, void \*value, size\_t maxLen);
* bool **writeBytes**(int address, const void \*value, size\_t len);

**Features:**

* void **init**(int size);

Description:

initialize the ram with the size being filled in

Parameters:

int *size* – fill a size of ram you want

Returns:

pointer+address

* uint16\_t **getSize**();

Description:

get the size of your ram

Returns:

byte uint16\_t size of ram

* uint8\_t **\*array**(int address);

Description:

add function

Parameters:

int *address* – fill a address of the ram you want

Returns:

pointer+address

* uint8\_t **readByte**(int address);

Description:

read 1-byte data at the filled address

Parameters:

int *address* – ram address you want to read

Returns:

the uint8\_t type value of the ram address you want to read

* bool **writeByte**(int address, uint8\_t value);

Description:

write 1-byte data at the filled address

Parameters:

int *address* – ram address you want to write

uint8\_t *value* – the value you want to write

Returns:

True/False

* int8\_t **readChar**(int address);

Description:

read 1- character data at the filled address

Parameters:

int *address* – ram address you want to read

Returns:

the int8\_t type value of the ram address you want to read

* bool **writeChar**(int address, int8\_t value);

Description:

write 1- character data at the filled address

Parameters:

int *address* – ram address you want to write

int8\_t *value* – the value you want to write

Returns:

True/False

* uint8\_t **readUChar**(int address);

Description:

read 1-unsigned character data at the filled address

Parameters:

int *address* – ram address you want to read

Returns:

the uint8\_t type value of the ram address you want to read

* bool **writeUChar**(int address, uint8\_t value);

Description:

write 1- unsigned character data at the filled address

Parameters:

int *address* – ram address you want to write

uint8\_t *value* – the value you want to write

Returns:

True/False

* int16\_t **readShort**(int address);

Description:

read 1-short data at the filled address

Parameters:

int *address* – ram address you want to read

Returns:

the int16\_t type value of the ram address you want to read

* bool **writeShort**(int address, int16\_t value);

Description:

write 1- short data at the filled address

Parameters:

int *address* – ram address you want to write

int16\_t *value* – the value you want to write

Returns:

True/False

* uint16\_t **readUShort**(int address);

Description:

read 1-unsigned short data at the filled address

Parameters:

int *address* – ram address you want to read

Returns:

the uint16\_t type value of the ram address you want to read

* bool **writeUShort**(int address, uint16\_t value);

Description:

write 1- unsigned short data at the filled address

Parameters:

int *address* – ram address you want to write

uint16\_t *value* – the value you want to write

Returns:

True/False

* int32\_t **readInt**(int address);

Description:

read 1-integer data at the filled address

Parameters:

int *address* – ram address you want to read

Returns:

the int32\_t type value of the ram address you want to read

* bool **writeInt**(int address, int32\_t value);

Description:

write 1- integer data at the filled address

Parameters:

int *address* – ram address you want to write

int32\_t *value* – the value you want to write

Returns:

True/False

* uint32\_t **readUInt**(int address);

Description:

read 1-unsigned integer data at the filled address

Parameters:

int *address* – ram address you want to read

Returns:

the uint32\_t type value of the ram address you want to read

* bool **writeUInt**(int address, uint32\_t value);

Description:

write 1- unsigned integer data at the filled address

Parameters:

int *address* – ram address you want to write

uint32\_t *value* – the value you want to write

Returns:

True/False

* int32\_t **readLong**(int address);

Description:

read 1-long data at the filled address

Parameters:

int *address* – ram address you want to read

Returns:

the int32\_t type value of the ram address you want to read

* bool **writeLong**(int address, int32\_t value);

Description:

write 1- long data at the filled address

Parameters:

int *address* – ram address you want to write

int32\_t *value* – the value you want to write

Returns:

True/False

* uint32\_t **readULong**(int address);

Description:

read 1-unsigned long data at the filled address

Parameters:

int *address* – ram address you want to read

Returns:

the uint32\_t type value of the ram address you want to read

* bool **writeULong**(int address, uint32\_t value);

Description:

write 1- unsigned long data at the filled address

Parameters:

int *address* – ram address you want to write

uint32\_t *value* – the value you want to write

Returns:

True/False

* int64\_t **readLong64**(int address);

Description:

read 1- long64 data at the filled address

Parameters:

int *address* – ram address you want to read

Returns:

the int64\_t type value of the ram address you want to read

* bool **writeLong64**(int address, int64\_t value);

Description:

write 1- long64 data at the filled address

Parameters:

int *address* – ram address you want to write

int64\_t *value* – the value you want to write

Returns:

True/False

* int64\_t **readULong64**(int address);

Description:

read 1- unsigned long64 data at the filled address

Parameters:

int *address* – ram address you want to read

Returns:

the int64\_t type value of the ram address you want to read

* bool **writeULong64**(int address, uint64\_t value);

Description:

write 1- unsigned long64 data at the filled address

Parameters:

int *address* – ram address you want to write

uint64\_t *value* – the value you want to write

Returns:

True/False

* float **readFloat**(int address);

Description:

read 1- float data at the filled address

Parameters:

int *address* – ram address you want to read

Returns:

the float type value of the ram address you want to read

* bool **writeFloat**(int address, float value);

Description:

write 1- float data at the filled address

Parameters:

int *address* – ram address you want to write

float *value* – the value you want to write

Returns:

True/False

* double **readDouble**(int address);

Description:

read 1- double data at the filled address

Parameters:

int *address* – ram address you want to read

Returns:

the double type value of the ram address you want to read

* bool **writeDouble**(int address, double value);

Description:

write 1- float data at the filled address

Parameters:

int *address* – ram address you want to write

double *value* – the value you want to write

Returns:

True/False

* bool **readBool**(int address);

Description:

read 1- boolean data at the filled address

Parameters:

int *address* – ram address you want to read

Returns:

the bool type value of the ram address you want to read

* bool **writeBool**(int address, bool value);

Description:

write 1- boolean data at the filled address

Parameters:

int *address* – ram address you want to write

bool *value* – the value you want to write

Returns:

True/False

* size\_t **readString**(int address, char \*value, size\_t maxLen);

Description:

copies data 1-string at the filled address to the pointer value in the form of an array

Parameters:

int *address* – ram address you want to copy

char \*value - array buffer

size\_t maxLen - the maximum length of the string

Returns:

the size\_t type length of the string

* bool **writeString**(int address, const char \*value);

Description:

copies data 1-string at the pointer value to the filled address in the form of an array

Parameters:

int *address* – ram address you want to write

const char \*value – array buffer

Returns:

True/False

* String **readString**(int address);

Description:

read 1- string data at the filled address

Parameters:

int *address* – ram address you want to read

Returns:

the string type value of the ram address you want to read

* bool **writeString**(int address, String value);

Description:

write 1- string data at the filled address

Parameters:

int *address* – ram address you want to write

String value – string you want to write

Returns:

True/False

* size\_t **readBytes**(int address, void \*value, size\_t maxLen);

Description:

Copies the values of *maxLen* bytes from the location pointed to by *value* directly to the memory block pointed to by *address*

Parameters:

int *address* – ram address you want to copy

void \*value – byte buffer

size\_t maxLen - the maximum length of the string

Returns:

the size\_t type length of the string

* bool **writeBytes**(int address, const void \*value, size\_t len);

Description:

Copies the values of *len* bytes from the location pointed to by *address* directly to the memory block pointed to by *value*

Parameters:

int *address* – ram address you want to write

String value – string you want to write

Returns:

True/False