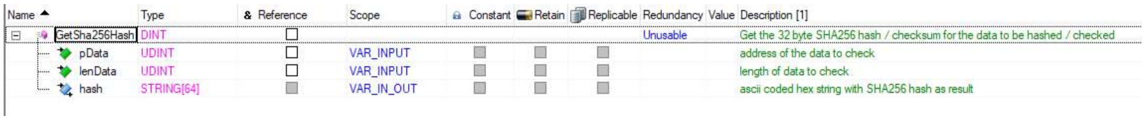
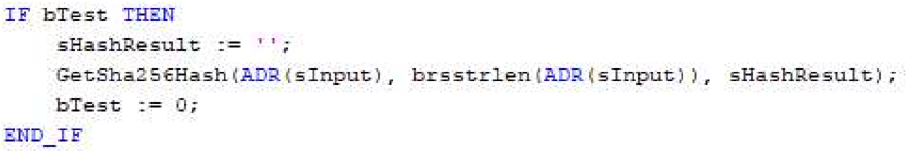
“Sha256lib” generates a SHA256 hash (“checksum”) over a data range (e.g., a string).
The hash is output as ASCII-encoded hexadecimal numbers .

Functional interface:

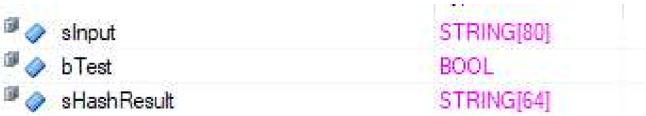


Name	Type	Reference	Scope	Constant	Retain	Replicable	Redundancy	Value	Description [1]
GetSha256Hash	DINT	□					Unusable		Get the 32 byte SHA256 hash / checksum for the data to be hashed / checked
pData	UDINT	□	VAR_INPUT	▨	▨	▨			address of the data to check
lenData	UDINT	□	VAR_INPUT	▨	▨	▨			length of data to check
hash	STRING[64]	▨	VAR_IN_OUT	▨	▨	▨			ascii coded hex string with SHA256 hash as result

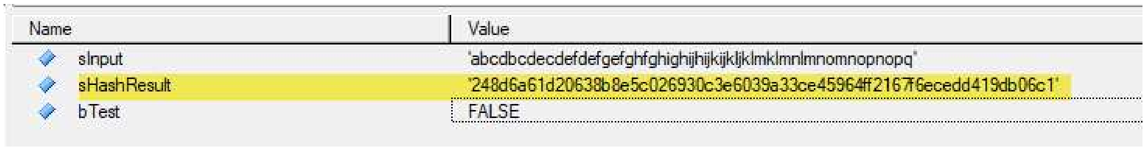
Example call:



```
IF bTest THEN
    sHashResult := '';
    GetSha256Hash(ADR(sInput), brsstrlen(ADR(sInput)), sHashResult);
    bTest := 0;
END_IF
```



sInput	STRING[80]
bTest	BOOL
sHashResult	STRING[64]

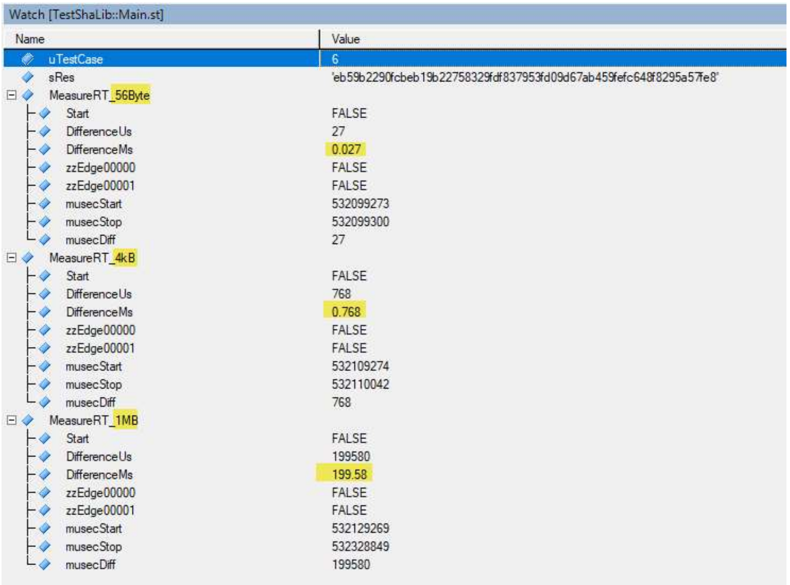


Name	Value
sInput	'abcdbcdecdefdefgefghfghighijhijkijkljklmklmnlmnomnopnopq'
sHashResult	'248d6a61d20638b8e5c026930c3e6039a33ce45964ff2167f6ecedd419db06c1'
bTest	FALSE

WARNING:

As with other crypto algorithms, such implementations are of course only conditionally suitable for the cyclic system - larger amounts of data should not be processed using them, due to the way the algorithms work (block formation in loops that cannot be designed "asynchronously" + therefore complete processing in one task cycle).

Test run with different data sizes on an X20P3585:
This results in a linear relationship between "data size and processing time", on the X20CP3585 for example approx. 0.19 - 0.2 microseconds per byte (net runtime, without further tasks / task classes).

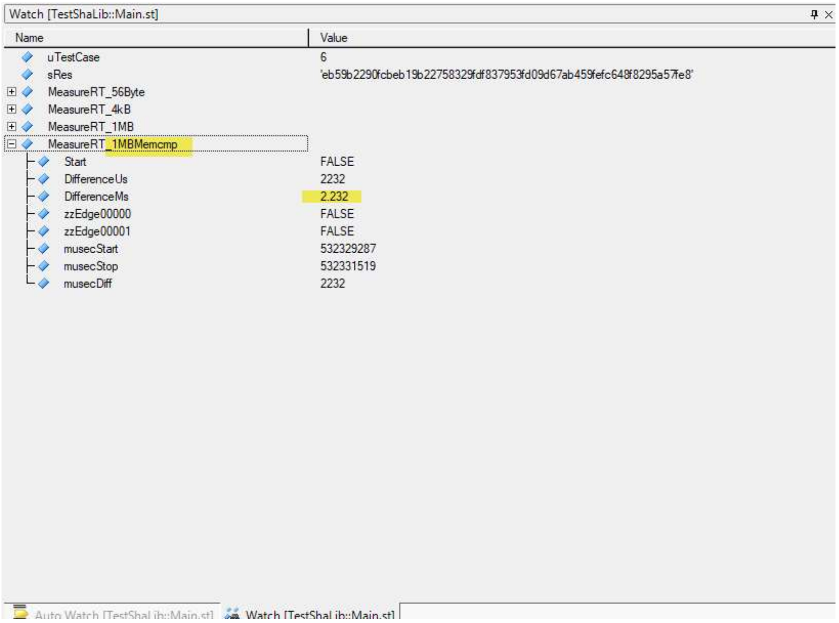


Watch [TestShaLib::Main.st]

Name	Value
uTestCase	6
sRes	'eb59b2290fcbeb19b22758329fdf837953fd09d67ab459fefc648f8295a57fe8'
MeasureRT_56Byte	
Start	FALSE
DifferenceUs	27
DifferenceMs	0.027
zzEdge00000	FALSE
zzEdge00001	FALSE
musecStart	532099273
musecStop	532099300
musecDiff	27
MeasureRT_4kB	
Start	FALSE
DifferenceUs	768
DifferenceMs	0.768
zzEdge00000	FALSE
zzEdge00001	FALSE
musecStart	532109274
musecStop	532110042
musecDiff	768
MeasureRT_1MB	
Start	FALSE
DifferenceUs	199580
DifferenceMs	199.58
zzEdge00000	FALSE
zzEdge00001	FALSE
musecStart	532129269
musecStop	532328849
musecDiff	199580

In comparison, the runtime of a brsmecmp ( ) call for approximately 1MB of data is approximately 2.232 milliseconds, compared to approximately 199.58 milliseconds for calculating the SHA256 checksum.

- ➔ This implementation of the checksum algorithm should not be used "to check memory contents for changes" when this can also be achieved by "making a copy of the memory and comparing it to the copy" (in which case, of course, twice the amount of memory is necessary!).



Watch [TestShaLib::Main.st]

Name	Value
uTestCase	6
sRes	'eb59b2290fcbeb19b22758329fdf837953fd09d67ab459fefc648f8295a57fe8'
MeasureRT_56Byte	
MeasureRT_4kB	
MeasureRT_1MB	
MeasureRT_1MBMemcmp	
Start	FALSE
DifferenceUs	2232
DifferenceMs	2.232
zzEdge00000	FALSE
zzEdge00001	FALSE
musecStart	532329287
musecStop	532331519
musecDiff	2232