

Lenze

Handbuch

IEC 61131-3

inside

***Global Drive
PLC Developer Studio***



Global Drive

Funktionsbibliothek

LenzeMemDrv.lib

Die Funktionsbibliothek **LenzeMemDrv.lib** kann für folgende Lenze PLC's verwendet werden:

	Typenbezeichnung	ab Hardwarestand	ab Softwarestand
9300 Servo PLC	EVS93XX-xI	6A	6.0
9300 Servo PLC	EVS93XX-xT	6A	6.0
Drive PLC	EPS10200	1A	6.0
ECSxA	ECSxAxxx	1C	7.0

Wichtiger Hinweis:

Die Software wird dem Benutzer in der vorliegenden Form zur Verfügung gestellt. Alle Risiken hinsichtlich der Qualität und der durch ihren Einsatz ermittelten Ergebnisse verbleiben beim Benutzer. Entsprechende Sicherheitsvorkehrungen gegen eventuelle Fehlbedienungen sind vom Benutzer vorzusehen.

Wir übernehmen keine Verantwortung für direkt oder indirekt entstandene Schäden, z. B. Gewinnverluste, Auftragsverluste oder geschäftliche Beeinträchtigungen jeglicher Art.

© 2002 Lenze Drive Systems GmbH

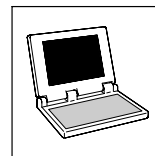
Ohne besondere schriftliche Genehmigung von Lenze Drive Systems GmbH darf kein Teil dieser Dokumentation vervielfältigt oder Dritten zugänglich gemacht werden.

Wir haben alle Angaben in dieser Dokumentation mit größter Sorgfalt zusammengestellt und auf Übereinstimmung mit der beschriebenen Hard- und Software geprüft. Trotzdem können wir Abweichungen nicht ganz ausschließen. Wir übernehmen keine juristische Verantwortung oder Haftung für Schäden, die dadurch eventuell entstehen. Notwendige Korrekturen werden wir in die nachfolgenden Auflagen einarbeiten.

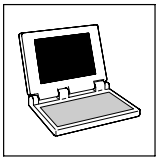
Windows, Windows NT und MS-DOS sind entweder eingetragene Warenzeichen oder Warenzeichen der Microsoft Corporation in den U.S.A und/oder anderen Ländern.

IBM und VGA sind eingetragene Warenzeichen von International Business Machines, Inc.

Alle anderen Markennamen sind Warenzeichen ihrer jeweiligen Besitzer.

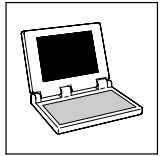


1 Vorwort und Allgemeines	1-1
1.1 Über dieses Handbuch	1-1
1.1.1 Konventionen in diesem Handbuch	1-1
1.1.2 Aufbau der Beschreibungen	1-2
1.1.3 Piktogramme in diesem Handbuch	1-2
1.1.4 Verwendete Begriffe	1-2
1.2 Versionskennung der Funktionsbibliothek	1-3
2 Einführung	2-1
2.1 Allgemeines zur Speicherverwaltung	2-1
2.2 Übersicht der Speicherverwaltungsfunktionen	2-1
2.2.1 Von Lenze PLC's unterstützte Funktionen	2-2
2.2.2 Hintergrundspeicher der Lenze PLC's	2-2
3 Allgemeine Speicherfunktionen	3-1
3.1 Übersicht	3-1
3.2 L_MemGetStateDriver	3-2
3.3 L_MemGetStateOfMemory	3-3
4 Funktionen für RAM-Speicherzugriffe	4-1
4.1 Übersicht	4-1
4.2 L_MemCopyFromRam	4-2
4.3 L_MemCopyToRam	4-4
4.4 L_MemReadDINT	4-6
4.5 L_MemReadDWORD	4-7
4.6 L_MemReadINT	4-8
4.7 L_MemReadWORD	4-9
4.8 L_MemWriteDINT	4-10
4.9 L_MemWriteDWORD	4-11
4.10 L_MemWriteINT	4-12
4.11 L_MemWriteWORD	4-13
5 Funktionen für FLASH-Speicherzugriffe	5-1
5.1 Übersicht	5-1
5.2 L_MemClearFlash	5-2
5.3 L_MemLoadRamFromFlash	5-3
5.4 L_MemSaveRamToFlash	5-5



Funktionsbibliothek LenzeMemDrv.lib

Inhalt



1 Vorwort und Allgemeines

1.1 Über dieses Handbuch

Dieses Handbuch enthält Informationen zur Funktionsbibliothek **LenzeMemDrv.lib** für das **Drive PLC Developer Studio**.

- In der Funktionsbibliothek **LenzeMemDrv.lib** sind Funktionen für den Schreib-/Lesezugriff auf den zusätzlichen Hintergrundspeicher (RAM, FLASH) der PLC enthalten.

1.1.1 Konventionen in diesem Handbuch

Dieses Handbuch verwendet folgende Konventionen zur Unterscheidung von verschiedenen Arten von Informationen:

Variablenbezeichner

sind im erklärenden Text in Kursivschrift dargestellt:

- “Mit *bReset...*”



Tip!

Informationen zu den Konventionen, die für die Variablenbezeichner von Lenze Systembausteinen, Funktionsblöcken sowie Funktionen verwendet werden, um eine einheitliche und durchgängige Benennung zu gewährleisten und dadurch die Lesbarkeit von SPS-Programmen zu fördern, finden Sie im Anhang der DDS-Online-Dokumentation “Einführung in die IEC1131-3 Programmierung”.

Lenze Funktionen/Funktionsblöcke

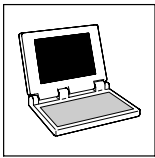
erkennen Sie am Namen. Sie beginnen immer mit einem “L_”:

- “Mit der Funktion **L_MemCopyToRam...**”

Programmlistings

sind in der Schriftart “Courier” aufgeführt, wobei Schlüsselwörter fett ausgezeichnet sind:

- “**IF** (ReturnValue < 0) **THEN...**”



Funktionsbibliothek LenzeMemDrv.lib

Vorwort und Allgemeines

1.1.2 Aufbau der Beschreibungen

Die Beschreibungen der einzelnen Funktionen/Funktionsblöcke sowie Systembausteine in diesem Handbuch sind einheitlich nach folgendem Schema aufgebaut:

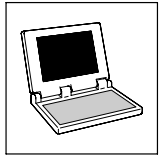
	Funktion	Funktionsblock (FB)/ Systembaustein (SB)
①	Überschrift mit Funktion und Bezeichner	
②	Deklaration der Funktion: - Datentyp des Rückgabewertes - Funktionsbezeichner - Liste der Übergabeparameter	-
③	Kurzbeschreibung mit den wichtigsten Eigenschaften	
④	Grafische Darstellung der Funktion mit den zugehörigen Variablen - Übergabeparameter - Rückgabewert	Grafische Darstellung des FB/SB mit den zugehörigen Variablen - Eingangsvariablen - Ausgangsvariablen
⑤	Tabelle mit Informationen zu den Übergabeparametern: - Bezeichner - Datentyp - Einstellmöglichkeiten - Info	Tabelle mit Informationen zu den Ein- und Ausgangsvariablen: - Bezeichner - Datentyp - Variablentyp - Einstellmöglichkeiten - Info
⑥	Tabelle mit Informationen zum Rückgabewert: - Datentyp des Rückgabewertes - Mögliche Rückgabewerte und ihre Bedeutung - Die Rückgabewerte sind in der Tabelle nach absteigender Priorität sortiert, d.h. der Rückgabewert mit der höchsten Priorität wird an erster Stelle aufgeführt.	-
⑦	Weitere Informationen (Anmerkungen, Tips, Anwendungsbeispiele, etc.)	

1.1.3 Piktogramme in diesem Handbuch

	verwendete Piktogramme	Signalwörter
Warnung vor Sachschäden		Stop! Warnt vor möglichen Sachschäden . Mögliche Folgen bei Mißachtung: Beschädigung des Antriebsreglers/Antriebssystems oder seiner Umgebung.
Sonstige Hinweise		Tip! Hinweis! Kennzeichnet einen Tip bzw. Hinweis.

1.1.4 Verwendete Begriffe

Begriff	Im folgenden Text verwendet für
DDS	Drive PLC Developer Studio
FB	Funktionsblock
GDC	Global Drive Control (Parametrier-Programm von Lenze)
Parameter-Codestellen	Codestellen zum Einstellen der Funktionalität eines Funktionsblocks
SB	Systembaustein



1.2 Versionskennung der Funktionsbibliothek

Die Version der Funktionsbibliothek können Sie über die globale Konstante `C_w[Funktionsbibliotheksname]Version` abfragen.

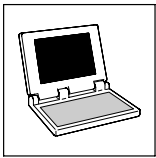
Versionskennung ab PLC Software-Stand 7.x:

Konstante	Bedeutung	Beispiel Wert
<code>C_w[Funktionsbibliotheksname]VersionER</code>	Externes Release	01
<code>C_w[Funktionsbibliotheksname]VersionEL</code>	Externes Level	05
<code>C_w[Funktionsbibliotheksname]VersionIR</code>	Internes Release	00
<code>C_w[Funktionsbibliotheksname]VersionBN</code>	Build-Nummer	00

Version: 01 05 00 00

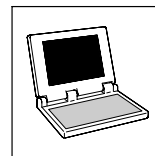
Der Wert dieser Konstanten ist hexadezimal-kodiert.

- Im Beispiel bedeutet der Wert "01050000" Version "1.05".



Funktionsbibliothek LenzeMemDrv.lib

Vorwort und Allgemeines



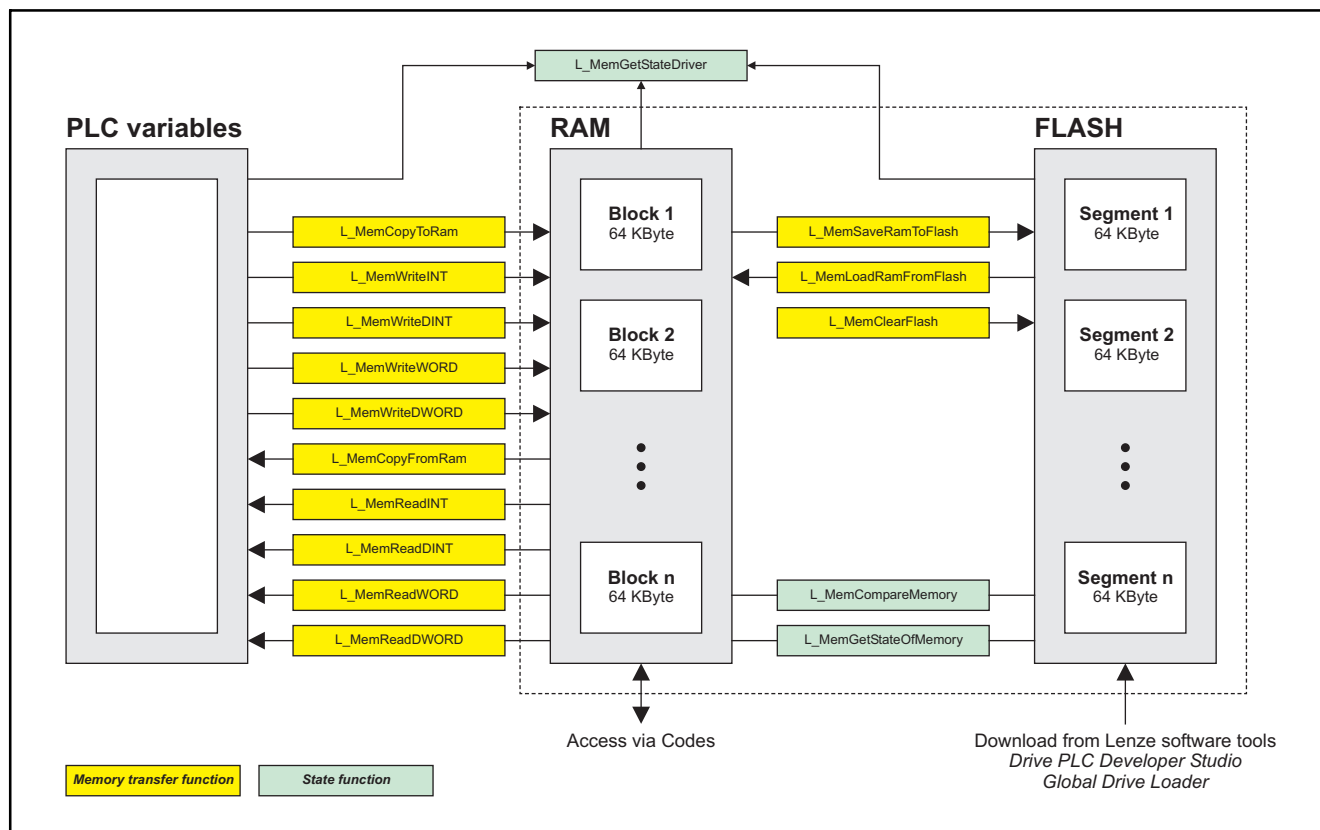
2 Einführung

2.1 Allgemeines zur Speicherverwaltung

Bei Lenze PLC's besteht die Möglichkeit, Teile des Variablenspeichers der PLC (z. B. globale Variablen, Arrays, usw.) in einen Hintergrundspeicher zu kopieren. Dadurch lassen sich z. B. Rezeptspeicher oder Meßwertspeicher realisieren.

2.2 Übersicht der Speicherverwaltungsfunktionen

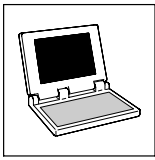
Eine Übersicht der zur Verfügung stehenden Funktionen zur Speicherverwaltung zeigt die folgende Abbildung:



Hinweis!

Die Art und Größe des zur Verfügung stehenden Hintergrundspeichers variiert bei den verschiedenen Lenze PLC's, daher können nicht alle Funktionen der **LenzeMemDrv.lib** für alle Lenze PLC's verwendet werden.

- Eine Gegenüberstellung, welche Lenze PLC welche Funktionen unterstützt, finden Sie im folgenden Unterkapitel.



Funktionsbibliothek LenzeMemDrv.lib

Einführung

Übersicht der Speicherverwaltungsfunktionen



Tip!

Zusätzlich zu den Funktionen der **LenzeMemDrv.lib** sind im Betriebssystem der PLC Codestellen implementiert, die u. a. folgende Speicher-relevanten Funktionen ermöglichen:

- Lesen/Schreiben beliebiger Daten eines RAM-Blocks mittels "Auto-Inkrementzugriff" während des laufenden SPS-Programms (C0504 ... C0508).
- Aktivieren eines Schreib-/Leseschutzes für den FLASH-Speicher (C2006/C2007).

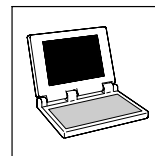
Ausführliche Informationen hierzu finden Sie im Handbuch zur entsprechenden PLC.

2.2.1 Von Lenze PLC's unterstützte Funktionen

	9300 Servo PLC	Drive PLC	ECSxA	
Allgemeine Speicherfunktionen				
L_MemGetStateDriver	✓	✓	✓	3-2
L_MemGetStateOfMemory	✓	✓	✓	3-3
Funktionen für RAM-Speicherzugriffe				
L_MemCopyFromRam	✓	✓	✓	4-2
L_MemCopyToRam	✓	✓	✓	4-4
L_MemReadDINT	✓	✓	✓	4-6
L_MemReadDWORD	✓	✓	✓	4-7
L_MemReadINT	✓	✓	✓	4-8
L_MemReadWORD	✓	✓	✓	4-9
L_MemWriteDINT	✓	✓	✓	4-10
L_MemWriteDWORD	✓	✓	✓	4-11
L_MemWriteWORD	✓	✓	✓	4-13
L_MemWriteINT	✓	✓	✓	4-12
Funktionen für FLASH-Speicherzugriffe				
L_MemClearFlash	✓			5-2
L_MemLoadRamFromFlash	✓			5-3
L_MemSaveRamToFlash	✓			5-5

2.2.2 Hintergrundspeicher der Lenze PLC's

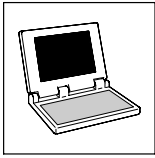
	9300 Servo PLC	Drive PLC	ECSxA
RAM	2 Blöcke á 64 KByte	2 Blöcke á 64 KByte	2 Blöcke á 64 KByte
FLASH	15 Segmente á 64 KByte	-	-



3 Allgemeine Speicherfunktionen

3.1 Übersicht

	9300 Servo PLC	Drive PLC	ECSxA	
L_MemGetStateDriver	✓	✓	✓	 3-2
L_MemGetStateOfMemory	✓	✓	✓	 3-3



Funktionsbibliothek LenzeMemDrv.lib

Allgemeine Speicherfunktionen

L_MemGetStateDriver

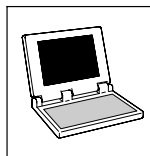
3.2 L_MemGetStateDriver

INT **L_MemGetStateDriver** (byVoid)

Diese Funktion liefert den aktuellen Treiberstatus.

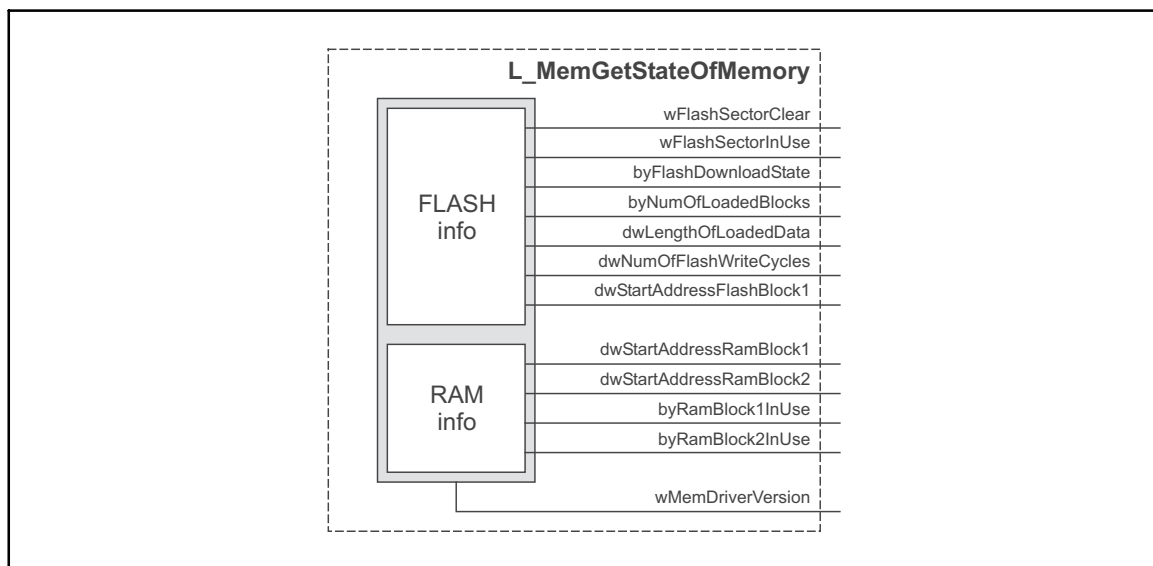
Übergabeparameter		
Bezeichner	Datentyp	Info/Einstellmöglichkeiten
byVoid	Byte	Keine Übergabeparameter

Rückgabewert: Integer		
Bit	Wert	Bedeutung
0	0	Treiber "CopyToRam" inaktiv
	1	Treiber "CopyToRam" aktiv
1	0	Treiber "CopyFromRam" inaktiv
	1	Treiber "CopyFromRam" aktiv
2	0	Treiber "LoadRamFromFlash" inaktiv
	1	Treiber "LoadRamFromFlash" aktiv
3	0	Treiber "SaveRamToFlash" inaktiv
	1	Treiber "SaveRamToFlash" aktiv
4	0	Treiber "ClearFlash" inaktiv
	1	Treiber "ClearFlash" aktiv

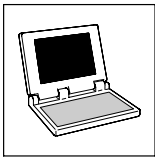


3.3 L_MemGetStateOfMemory

Mit diesem FB kann der Speicherstatus der PLC abgefragt werden.



Ausgänge ⇨	(Variablentyp: VAR_OUTPUT)		
Bezeichner	Datentyp	Wert	Bedeutung
wFlashSectorClear	Word	Bit:	
		0	0 Flash-Speichersegment 1 ist nicht gelöscht.
			1 Flash-Speichersegment 1 ist gelöscht.
		1	0 Flash-Speichersegment 2 ist nicht gelöscht.
			1 Flash-Speichersegment 2 ist gelöscht.
		...	0 Flash-Speichersegment 3 ... 14 ist nicht gelöscht.
			1 Flash-Speichersegment 3 ... 14 ist gelöscht.
		14	0 Flash-Speichersegment 15 ist nicht gelöscht.
wFlashSectorInUse	Word		1 Flash-Speichersegment 15 ist gelöscht.
		15	Wird intern vom Betriebssystem verwendet.
		Bit:	
		0	0 Flash-Speichersegment 1 wird momentan nicht beschrieben.
			1 Flash-Speichersegment 1 wird momentan beschrieben.
		1	0 Flash-Speichersegment 2 wird momentan nicht beschrieben.
			1 Flash-Speichersegment 2 wird momentan beschrieben.
		...	0 Flash-Speichersegment 3 ... 14 wird momentan nicht beschrieben.
byFlashDownloadState	Byte		1 Flash-Speichersegment 3 ... 14 wird momentan beschrieben.
		14	0 Flash-Speichersegment 15 wird momentan nicht beschrieben.
			1 Flash-Speichersegment 15 wird momentan beschrieben.
byNumOfLoadedBlocks	Byte	15	Wird intern vom Betriebssystem verwendet.
			0 Flash gelöscht.
			1 Flash-Download wurde durchgeführt, keine Änderungen.
dwLengthOfLoadedData	Double Word		2 Flash-Download wurde nicht erfolgreich beendet ("Timeout Communication Error").
dwStartAddressFlashBlock1	Double Word	1 ... 15	Anzahl der Flash-Speichersegmente, bei denen ein Download durchgeführt wurde.
		1 ... 983040	Anzahl der in den Flash-Speicher geladenen Bytes
			Startadresse des Flash-Speichersegmentes 1



Funktionsbibliothek LenzeMemDrv.lib

Allgemeine Speicherfunktionen

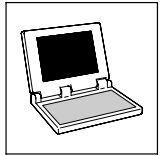
L_MemGetStateOfMemory

Ausgänge ⇨	(Variablentyp: VAR_OUTPUT)		
Bezeichner	Datentyp	Wert	Bedeutung
dwStartAddress RamBlock1	Double Word		Startadresse des RAM-Speicherblocks 1
dwStartAddress RamBlock2	Double Word		Startadresse des RAM-Speicherblocks 2
byRamBlock1InUse	Byte	0	RAM-Speicherblock 1 wird momentan nicht beschrieben.
		1	RAM-Speicherblock 1 wird momentan beschrieben.
byRamBlock2InUse	Byte	0	RAM-Speicherblock 2 wird momentan nicht beschrieben.
		1	RAM-Speicherblock 2 wird momentan beschrieben.
wMemDriverVersion	Word		Version des Treibers für den Speicherzugriff Der Wert dieser Konstante ist hex-codiert. Der Wert "16#0130" bedeutet z. B. "Version 1.30".



Tip!

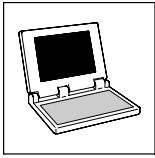
Der aktuelle Speicherstatus wird vom Betriebssystem fortlaufend aktualisiert, er ändert sich z. B. durch den Aufruf der Funktion **L_MemCopyToRam**.



4 Funktionen für RAM-Speicherzugriffe

4.1 Übersicht

	9300 Servo PLC	Drive PLC	ECSxA	
L_MemCopyFromRam	✓	✓	✓	4-2
L_MemCopyToRam	✓	✓	✓	4-4
L_MemReadDINT	✓	✓	✓	4-6
L_MemReadDWORD	✓	✓	✓	4-7
L_MemReadINT	✓	✓	✓	4-8
L_MemReadWORD	✓	✓	✓	4-9
L_MemWriteDINT	✓	✓	✓	4-10
L_MemWriteDWORD	✓	✓	✓	4-11
L_MemWriteWORD	✓	✓	✓	4-13
L_MemWriteINT	✓	✓	✓	4-12



Funktionsbibliothek LenzeMemDrv.lib

Funktionen für RAM-Speicherzugriffe

L_MemCopyFromRam

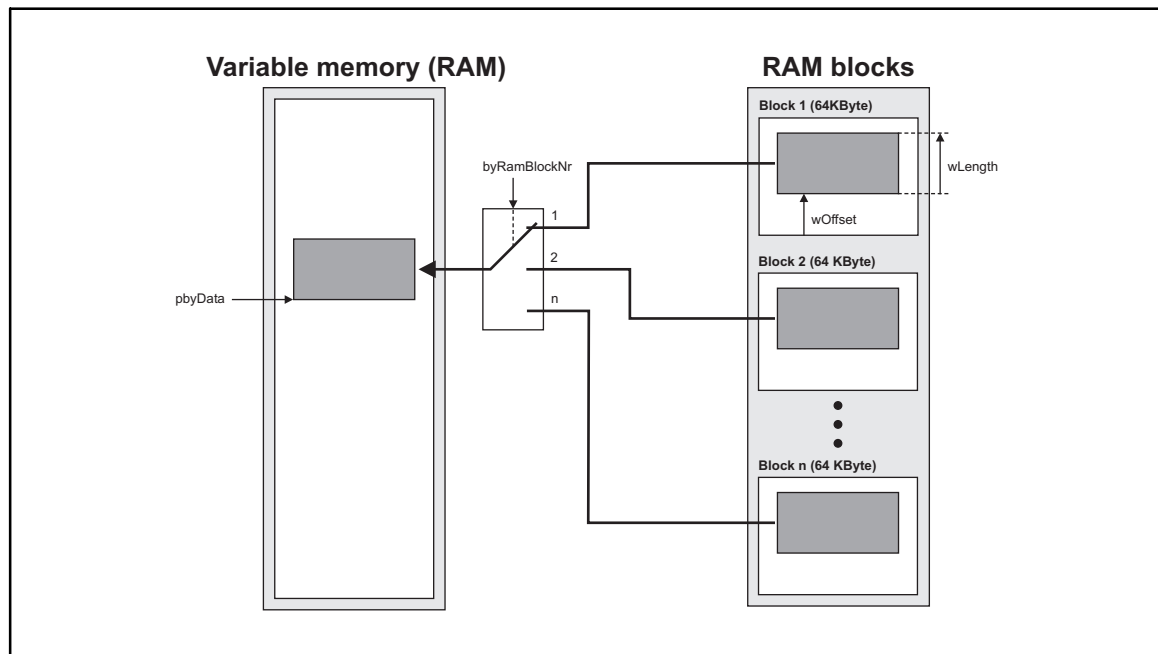
4.2

L_MemCopyFromRam

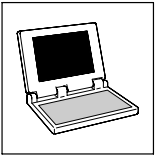
INT **L_MemCopyFromRam**(byRamBlockNr, wOffset, pbyData, wLength)

Mit dieser Funktion können Daten aus einem RAM-Block der PLC bytewise in den Variablen-Speicher der PLC kopiert werden.

- Die Funktion wird parallel zum PLC-Programm in der Systemtask abgearbeitet, die Bearbeitungsdauer ist daher abhängig von der Auslastung des Systems.
- Der Abschluß des Kopiervorgangs kann mit der Funktion **L_MemGetStateDriver** ermittelt werden.



Übergabeparameter		
Bezeichner	Datentyp	Info/Einstellmöglichkeiten
byRamBlockNr	Byte	Auswahl des RAM-Speicherblocks, in dem sich die zu kopierenden Datenbytes befinden
		1 RAM-Speicher 1, Größe: 64 KByte
		2 RAM-Speicher 2, Größe: 64 KByte
wOffset	Word	Startadresse im RAM-Speicherblock für die zu kopierenden Datenbytes
		Die Startadresse darf nicht größer sein als 65535 - wLength.
pbyData	Pointer to Byte	Zeiger auf den Zielpunkt im Variablen-Speicher der PLC für die Ablage der zu kopierenden Datenbytes
wLength	Word	1 ... 8192 Anzahl der zu kopierenden Datenbytes



Rückgabewert: Integer	
Wert	Bedeutung
0	Kopierauftrag wurde erfolgreich an das Betriebssystem übergeben.
-10	Kopierauftrag wurde nicht an das Betriebssystem übergeben, da Übergabeparameter fehlerhaft: <i>wLength</i> = 0.
-20	Kopierauftrag wurde nicht an das Betriebssystem übergeben, da Übergabeparameter fehlerhaft: <i>pbyData</i> zeigt nicht auf den Speicherbereich der SPS.
-30	Kopierauftrag wurde nicht an das Betriebssystem übergeben, da Übergabeparameter fehlerhaft: Die Angaben von <i>wLength</i> und <i>wOffset</i> sind nicht plausibel zueinander.
-40	Kopierauftrag wurde nicht an das Betriebssystem übergeben, da eine zuvor gestartete Speicherverwaltungsfunktion noch den Kopiertreiber belegt.
-50	Kopierauftrag wurde nicht an das Betriebssystem übergeben, da zur Zeit ein Zugriff auf den Speicher stattfindet (z. B. über Codestellen).
-60	Kopierauftrag wurde nicht an das Betriebssystem übergeben, da Übergabeparameter fehlerhaft: <i>byRamBlockNr</i> ist ungültig.
-80	Funktion wird vom Betriebssystem nicht unterstützt.

Beispiel

Aufruf der Funktion in ST:

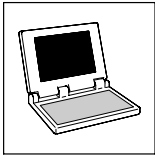
```
(* copy from RAM if last copy action (L_MemCopyFromRam) *)
(* is finished *)

nMemState := L_MemGetStateDriver(byVoid := 0);
bDriverBusy := INT_TO_BOOL(nMemState AND 16#0002);

IF bDriverBusy = FALSE THEN
    nReturn := L_MemCopyFromRam(byRamBlockNr := 2,
                                wOffset := 512,
                                pbyData := ADR(g_abyDataBuffer),
                                wLength := 256);

END_IF

(* check return value of CopyFromRam function *)
```



Funktionsbibliothek LenzeMemDrv.lib

Funktionen für RAM-Speicherzugriffe

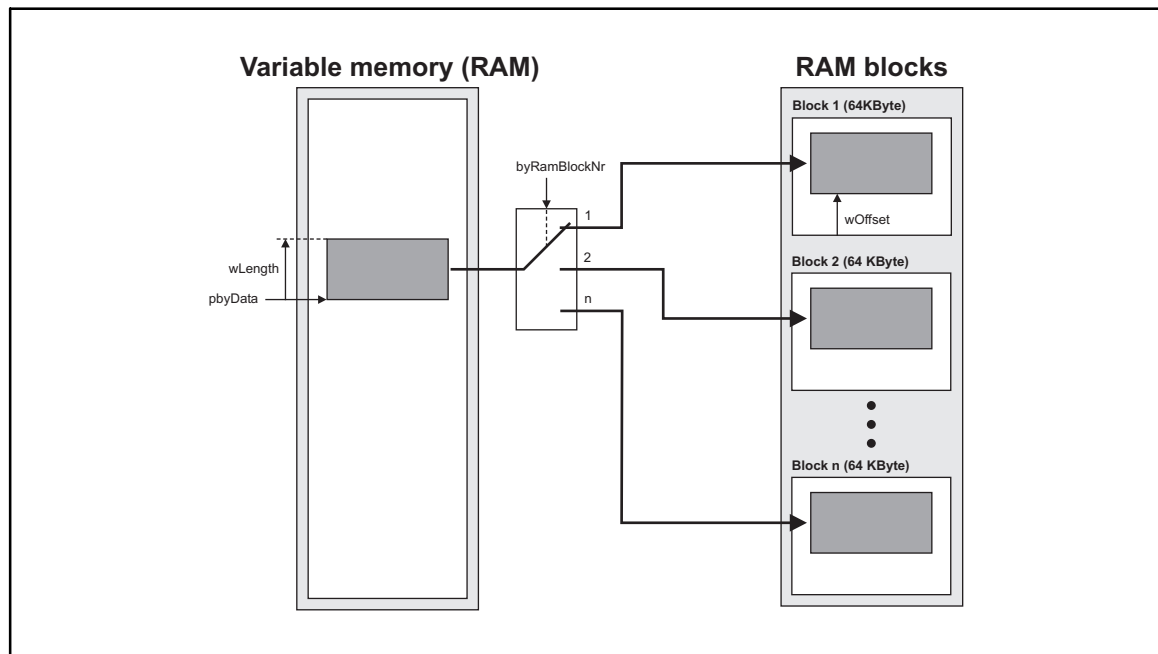
L_MemCopyToRam

4.3 L_MemCopyToRam

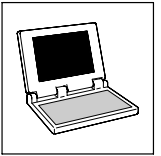
INT **L_MemCopyToRam**(pbyData, wLength, byRamBlockNr, wOffset)

Mit dieser Funktion können Daten aus dem Variablen-Speicher der PLC byteweise in einen RAM-Block der PLC kopiert werden.

- Die Funktion wird parallel zum PLC-Programm in der Systemtask abgearbeitet, die Bearbeitungsdauer ist daher abhängig von der Auslastung des Systems.
- Der Abschluß des Kopiervorgangs kann mit der Funktion **L_MemGetStateDriver** ermittelt werden.



Übergabeparameter		
Bezeichner	Datentyp	Info/Einstellmöglichkeiten
pbyData	Pointer to Byte	Zeiger auf den Startpunkt im Variablen-Speicher der PLC, ab dem sich die zu kopierenden Datenbytes befinden
wLength	Word	1 ... 8192 Anzahl der zu kopierenden Datenbytes
byRamBlockNr	Byte	Auswahl des RAM-Speicherblocks, in den die Datenbytes kopiert werden sollen
		1 RAM-Speicher 1, Größe: 64 KByte
		2 RAM-Speicher 2, Größe: 64 KByte
wOffset	Word	Zielposition innerhalb des ausgewählten RAM-Speicherblocks, welche die Anfangsadresse für die Ablage der zu kopierenden Datenbytes darstellt. Die Zielposition darf nicht größer sein als 65535 - wLength.
		0 ... 65535 Zielposition (Offset)



Rückgabewert: Integer	
Wert	Bedeutung
0	Kopierauftrag wurde erfolgreich an das Betriebssystem übergeben.
-10	Kopierauftrag wurde nicht an das Betriebssystem übergeben, da Übergabeparameter fehlerhaft: <i>wLength</i> = 0.
-20	Kopierauftrag wurde nicht an das Betriebssystem übergeben, da Übergabeparameter fehlerhaft: <i>pbyData</i> zeigt nicht auf den Speicherbereich der SPS.
-30	Kopierauftrag wurde nicht an das Betriebssystem übergeben, da Übergabeparameter fehlerhaft: Die Angaben von <i>wLength</i> und <i>wOffset</i> sind nicht plausibel zueinander.
-40	Kopierauftrag wurde nicht an das Betriebssystem übergeben, da eine zuvor gestartete Speicherverwaltungsfunktion noch den Kopiertreiber belegt.
-50	Kopierauftrag wurde nicht an das Betriebssystem übergeben, da zur Zeit ein Zugriff auf den Speicher stattfindet (z. B. über Codestellen).
-60	Kopierauftrag wurde nicht an das Betriebssystem übergeben, da Übergabeparameter fehlerhaft: <i>byRamBlockNr</i> ist ungültig.
-80	Funktion wird vom Betriebssystem nicht unterstützt.

Beispiel

Aufruf der Funktion in ST:

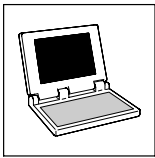
```
(* copy to RAM if last copy action (L_MemCopyToRam) *)
(* is finished *)

nMemState := L_MemGetStateDriver(byVoid := 0);
bDriverBusy := INT_TO_BOOL(nMemState AND 16#0001);

IF bDriverBusy = FALSE THEN
    nReturn := L_MemCopyToRam(pbyData := ADR(g_abyDataBuffer),
                               wLength := 256,
                               byRamBlockNr := 2,
                               wOffset := 512);

END_IF

(* check return value of CopyToRam function *)
```

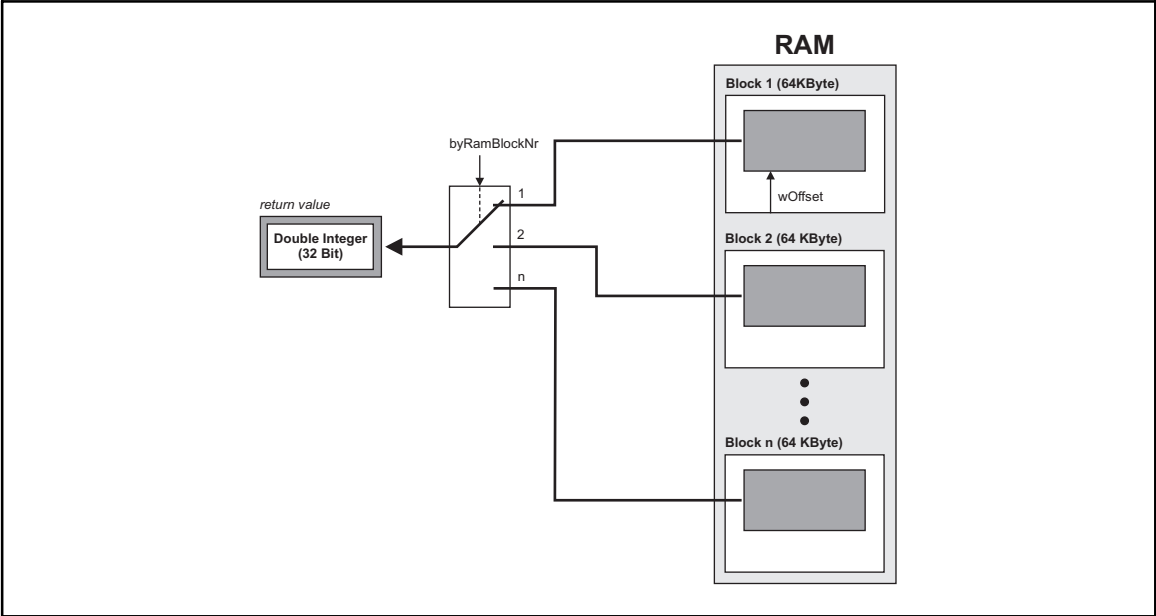


4.4

L_MemReadDINT

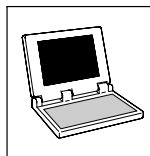
DINT
L_MemReadDINT
(byRamBlockNr, wOffset)

Mit dieser Funktion kann ein 32-Bit-Wert vom Typ "Double Integer" aus einem RAM-Block der PLC gelesen werden.



Übergabeparameter		
Bezeichner	Datentyp	Info/Einstellmöglichkeiten
byRamBlockNr	Byte	Auswahl des RAM-Speicherblocks, aus dem der 32-Bit-Wert gelesen werden soll.
		1 RAM-Speicher 1, Größe: 64 KByte
		2 RAM-Speicher 2, Größe: 64 KByte
wOffset	Word	Startposition innerhalb des ausgewählten RAM-Speicherblocks, von der der 16-Bit-Wert gelesen werden soll.
		0 ... 65532 Startposition (Offset)

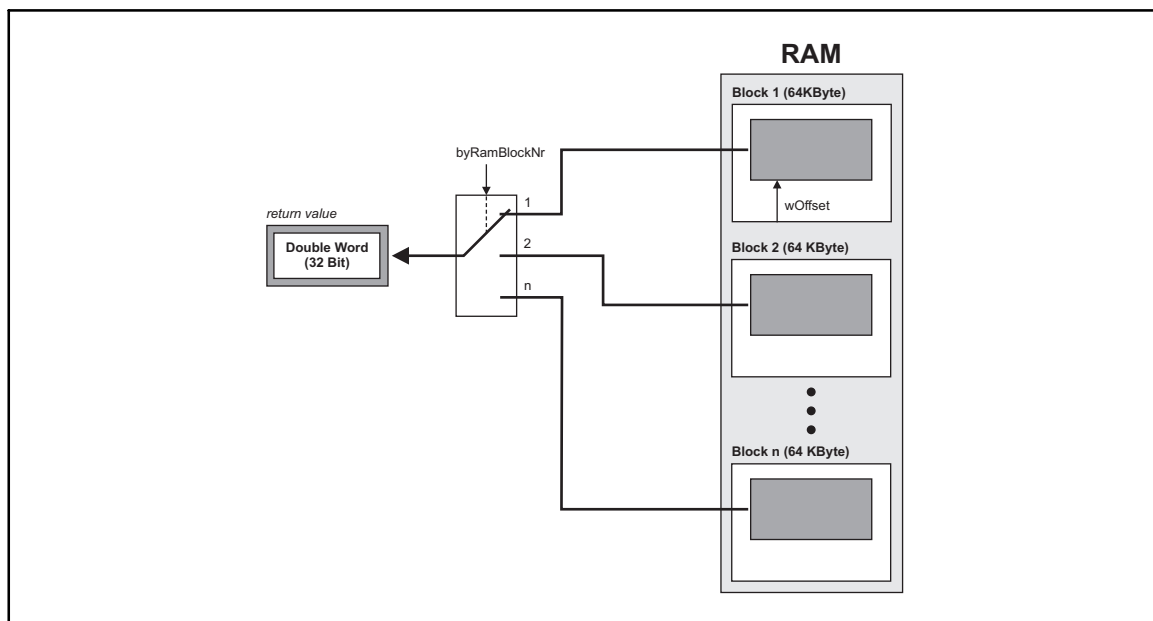
Rückgabewert: Double Integer	
Wert	Bedeutung
Daten	32-Bit-Wert, der im RAM-Speicherblock <i>byRamBlockNr</i> von der Position <i>wOffset</i> gelesen wurde.
NULL	Leseauftrag wurde nicht an das Betriebssystem übergeben, da Übergabeparameter fehlerhaft: <i>byRamBlockNr</i> ist ungültig.



4.5 L_MemReadDWORD

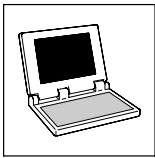
DWORD **L_MemReadDWORD** (byRamBlockNr, wOffset)

Mit dieser Funktion kann ein 32-Bit-Wert vom Typ "Double Word" aus einem RAM-Block der PLC gelesen werden.



Übergabeparameter		
Bezeichner	Datentyp	Info/Einstellmöglichkeiten
byRamBlockNr	Byte	Auswahl des RAM-Speicherblocks, aus dem der 32-Bit-Wert gelesen werden soll.
		1 RAM-Speicher 1, Größe: 64 KByte
		2 RAM-Speicher 2, Größe: 64 KByte
wOffset	Word	Startposition innerhalb des ausgewählten RAM-Speicherblocks, von der der 32-Bit-Wert gelesen werden soll.
		0 ... 65532 Startposition (Offset)

Rückgabewert: Double Word	
Wert	Bedeutung
Daten	32-Bit-Wert, der im RAM-Speicherblock <i>byRamBlockNr</i> von der Position <i>wOffset</i> gelesen wurde.
NULL	Leseauftrag wurde nicht an das Betriebssystem übergeben, da Übergabeparameter fehlerhaft: <i>byRamBlockNr</i> ist ungültig.

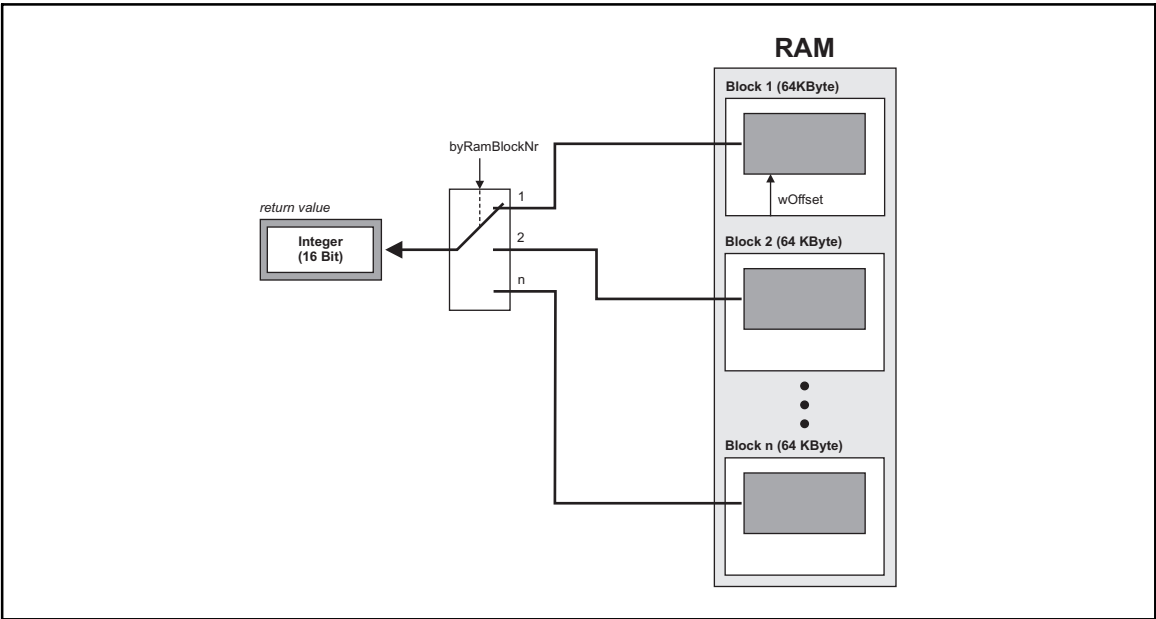


4.6

L_MemReadINT

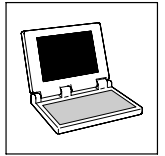
```
INT L_MemReadINT (byRamBlockNr, wOffset)
```

Mit dieser Funktion kann ein 16-Bit-Wert vom Typ "Integer" aus einem RAM-Block der PLC gelesen werden.



Übergabeparameter		
Bezeichner	Datentyp	Info/Einstellmöglichkeiten
byRamBlockNr	Byte	Auswahl des RAM-Speicherblocks, aus dem der 16-Bit-Wert gelesen werden soll.
		1 RAM-Speicher 1, Größe: 64 KByte
		2 RAM-Speicher 2, Größe: 64 KByte
wOffset	Word	Startposition innerhalb des ausgewählten RAM-Speicherblocks, von der der 16-Bit-Wert gelesen werden soll.
		0 ... 65534 Startposition (Offset)

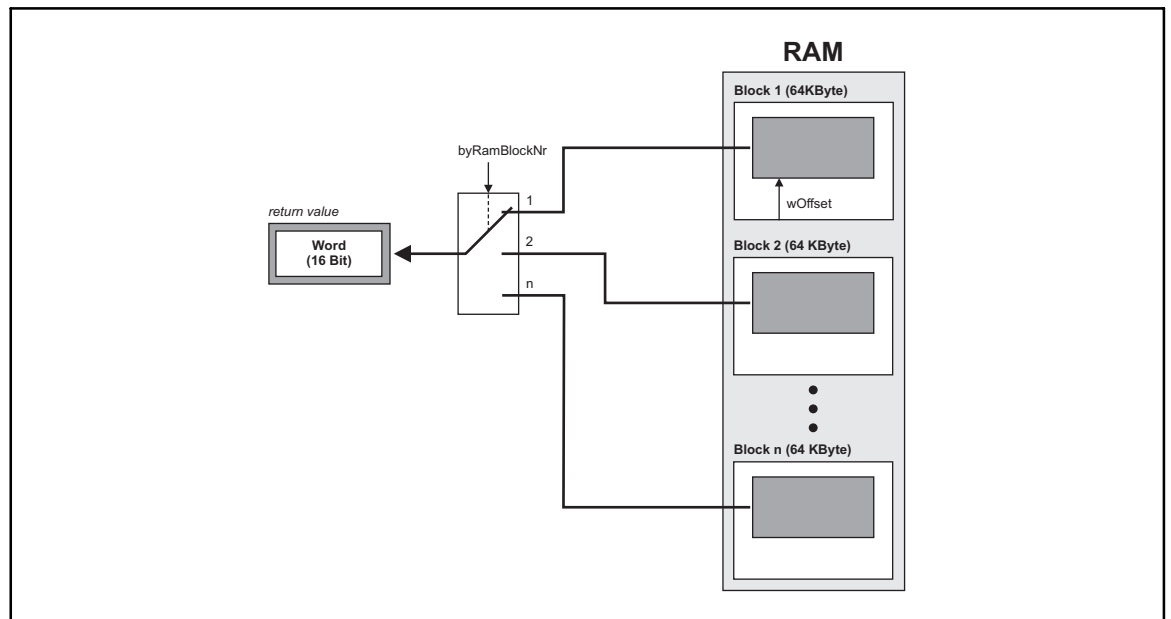
Rückgabewert: Integer	
Wert	Bedeutung
Daten	16-Bit-Wert, der im RAM-Speicherblock <i>byRamBlockNr</i> von der Position <i>wOffset</i> gelesen wurde.
NULL	Leseauftrag wurde nicht an das Betriebssystem übergeben, da Übergabeparameter fehlerhaft: <i>byRamBlockNr</i> ist ungültig.



4.7 L_MemReadWORD

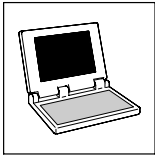
WORD **L_MemReadWORD** (byRamBlockNr, wOffset)

Mit dieser Funktion kann ein 16-Bit-Wert vom Typ "Word" aus einem RAM-Block der PLC gelesen werden.



Übergabeparameter		
Bezeichner	Datentyp	Info/Einstellmöglichkeiten
byRamBlockNr	Byte	Auswahl des RAM-Speicherblocks, aus dem der 16-Bit-Wert gelesen werden soll.
		1 RAM-Speicher 1, Größe: 64 KByte
		2 RAM-Speicher 2, Größe: 64 KByte
wOffset	Word	Startposition innerhalb des ausgewählten RAM-Speicherblocks, von der der 16-Bit-Wert gelesen werden soll.
		0 ... 65534 Startposition (Offset)

Rückgabewert: Word	
Wert	Bedeutung
Daten	16-Bit-Wert, der im RAM-Speicherblock <i>byRamBlockNr</i> von der Position <i>wOffset</i> gelesen wurde.
NULL	Leseauftrag wurde nicht an das Betriebssystem übergeben, da Übergabeparameter fehlerhaft: <i>byRamBlockNr</i> ist ungültig.



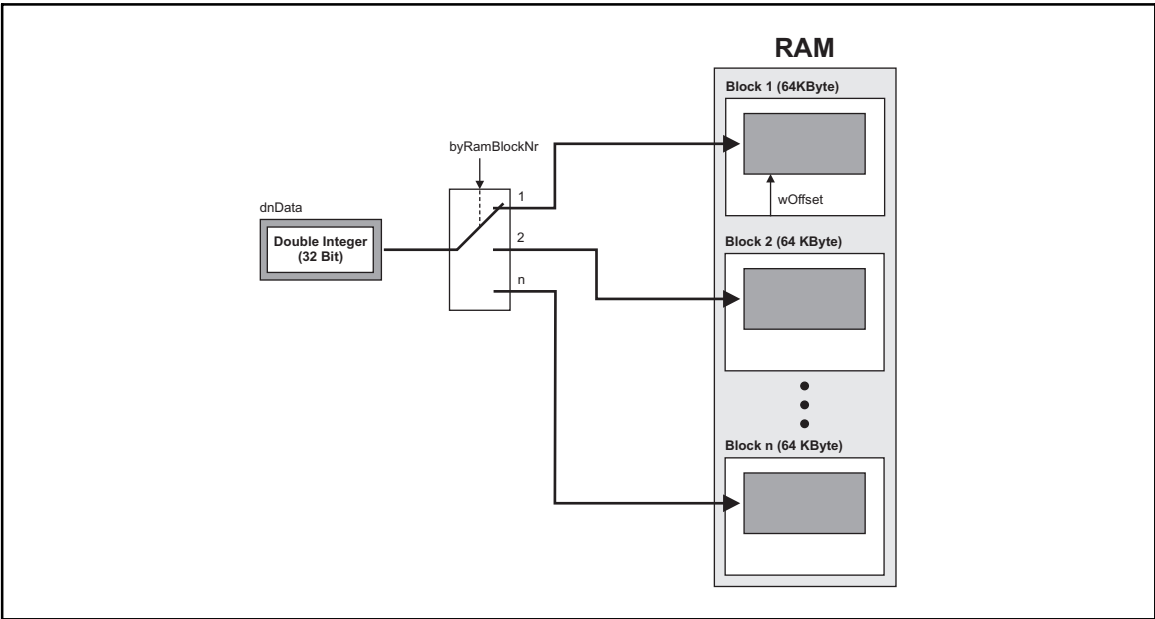
Funktionsbibliothek LenzeMemDrv.lib

Funktionen für RAM-Speicherzugriffe L_MemWriteDINT

4.8 L_MemWriteDINT

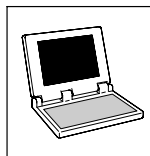
`INT L_MemWriteDINT (dnData, byRamBlockNr, wOffset)`

Mit dieser Funktion kann der Wert einer Variable vom Typ "Double Integer" in einen RAM-Block der PLC geschrieben werden.



Übergabeparameter		
Bezeichner	Datentyp	Info/Einstellmöglichkeiten
dnData	Double Integer	Zu schreibender Datenwert (32 Bit).
byRamBlockNr	Byte	Auswahl des RAM-Speicherblocks, in den <i>dnData</i> geschrieben werden soll.
		1 RAM-Speicher 1, Größe: 64 KByte
		2 RAM-Speicher 2, Größe: 64 KByte
wOffset	Word	Zielposition innerhalb des ausgewählten RAM-Speicherblocks, an die <i>dnData</i> geschrieben werden soll.
		0 ... 65532 Zielposition (Offset) • Als Offset sind nur gerade Werte zulässig!

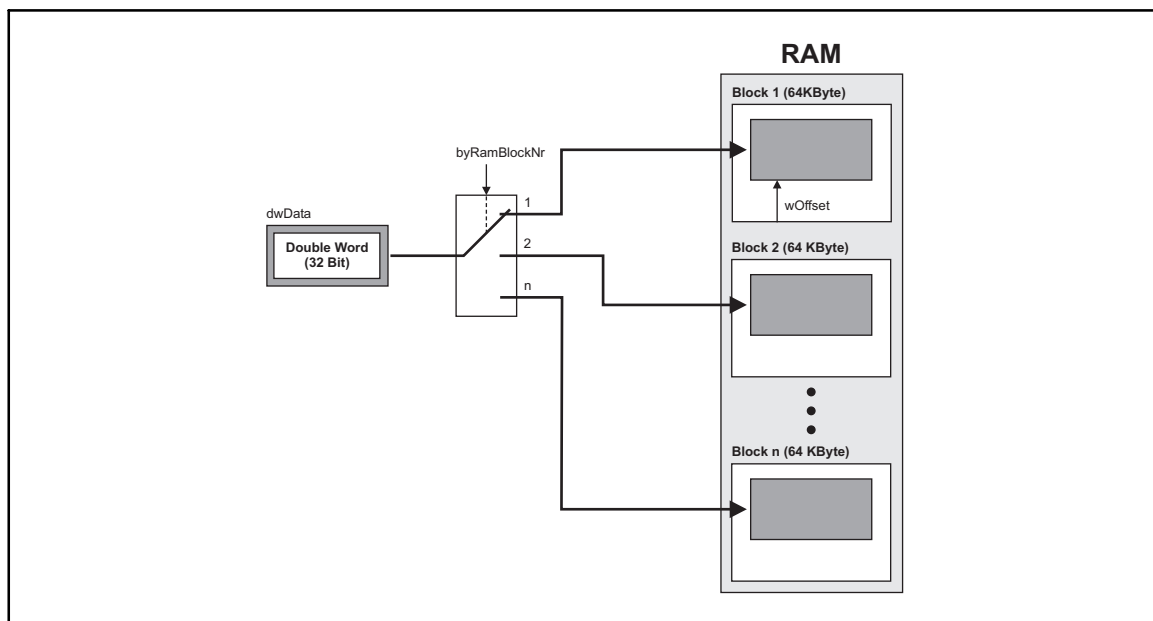
Rückgabewert: Integer	
Wert	Bedeutung
0	Schreibauftrag ist abgeschlossen.
-60	Schreibauftrag wurde nicht an das Betriebssystem übergeben, da Übergabeparameter fehlerhaft: <i>byRamBlockNr</i> ist ungültig.
-35	Kopierauftrag wurde nicht an das Betriebssystem übergeben, da Übergabeparameter fehlerhaft: <i>wOffset</i> ist eine ungerade Adresse oder Wert ist außerhalb des zulässigen Bereichs (0 ... 65532).
-50	Kopierauftrag wurde nicht an das Betriebssystem übergeben, da zur Zeit ein Zugriff auf den Speicher stattfindet (z. B. über Codestellen).



4.9 L_MemWriteDWORD

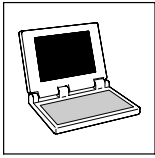
INT **L_MemWriteDWORD**(dwData, byRamBlockNr, wOffset)

Mit dieser Funktion kann der Wert einer Variable vom Typ "Double Word" in einen RAM-Block der PLC geschrieben werden.



Übergabeparameter		
Bezeichner	Datentyp	Info/Einstellmöglichkeiten
dwData	Double Word	Zu schreibender Datenwert (32 Bit).
byRamBlockNr	Byte	Auswahl des RAM-Speicherblocks, in den <i>dwData</i> geschrieben werden soll.
		1 RAM-Speicher 1, Größe: 64 KByte
		2 RAM-Speicher 2, Größe: 64 KByte
wOffset	Word	Zielposition innerhalb des ausgewählten RAM-Speicherblocks, an die <i>dwData</i> geschrieben werden soll.
		0 ... 65532 Zielposition (Offset) • Als Offset sind nur gerade Werte zulässig!

Rückgabewert: Integer	
Wert	Bedeutung
0	Schreibauftrag ist abgeschlossen.
-60	Schreibauftrag wurde nicht an das Betriebssystem übergeben, da Übergabeparameter fehlerhaft: <i>byRamBlockNr</i> ist ungültig.
-35	Kopierauftrag wurde nicht an das Betriebssystem übergeben, da Übergabeparameter fehlerhaft: <i>wOffset</i> ist eine ungerade Adresse oder Wert ist außerhalb des zulässigen Bereichs (0 ... 65532).
-50	Kopierauftrag wurde nicht an das Betriebssystem übergeben, da zur Zeit ein Zugriff auf den Speicher stattfindet (z. B. über Codestellen).



Funktionsbibliothek LenzeMemDrv.lib

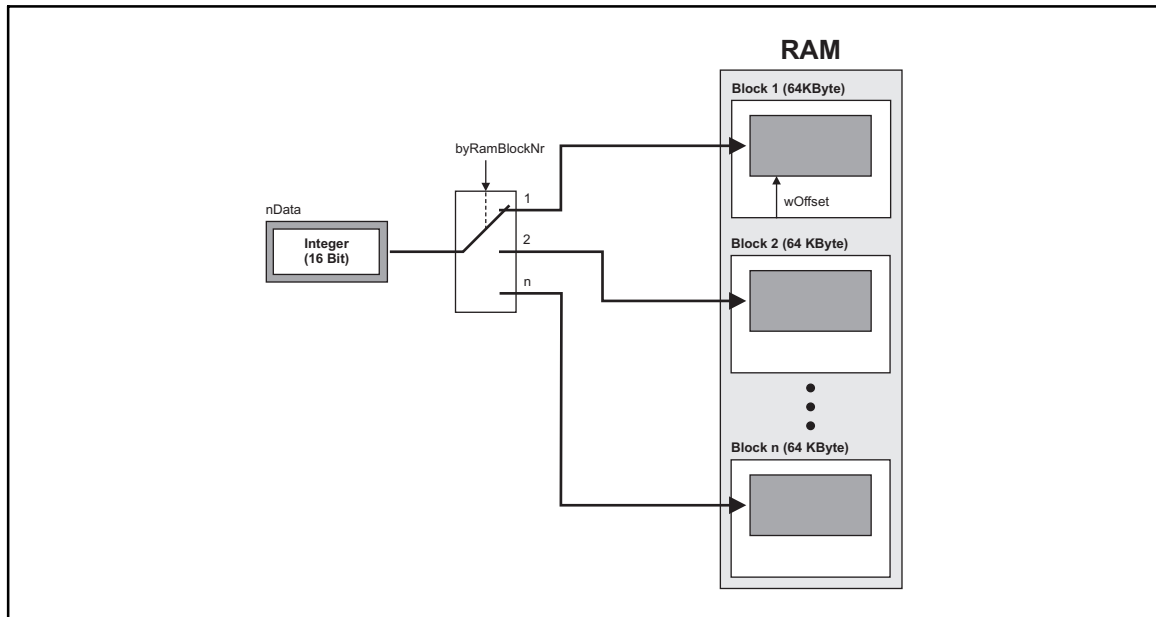
Funktionen für RAM-Speicherzugriffe

L_MemWriteINT

4.10 L_MemWriteINT

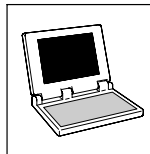
INT **L_MemWriteINT**(nData, byRamBlockNr, wOffset)

Mit dieser Funktion kann der Wert einer Variable vom Typ "Integer" in einen RAM-Block der PLC geschrieben werden.



Übergabeparameter		
Bezeichner	Datentyp	Info/Einstellmöglichkeiten
nData	Integer	Zu schreibender Datenwert (16 Bit).
byRamBlockNr	Byte	Auswahl des RAM-Speicherblocks, in den nData geschrieben werden soll.
		1 RAM-Speicher 1, Größe: 64 KByte
		2 RAM-Speicher 2, Größe: 64 KByte
wOffset	Word	Zielposition innerhalb des ausgewählten RAM-Speicherblocks, an die nData geschrieben werden soll.
		0 ... 65534 Zielposition (Offset) • Als Offset sind nur gerade Werte zulässig!

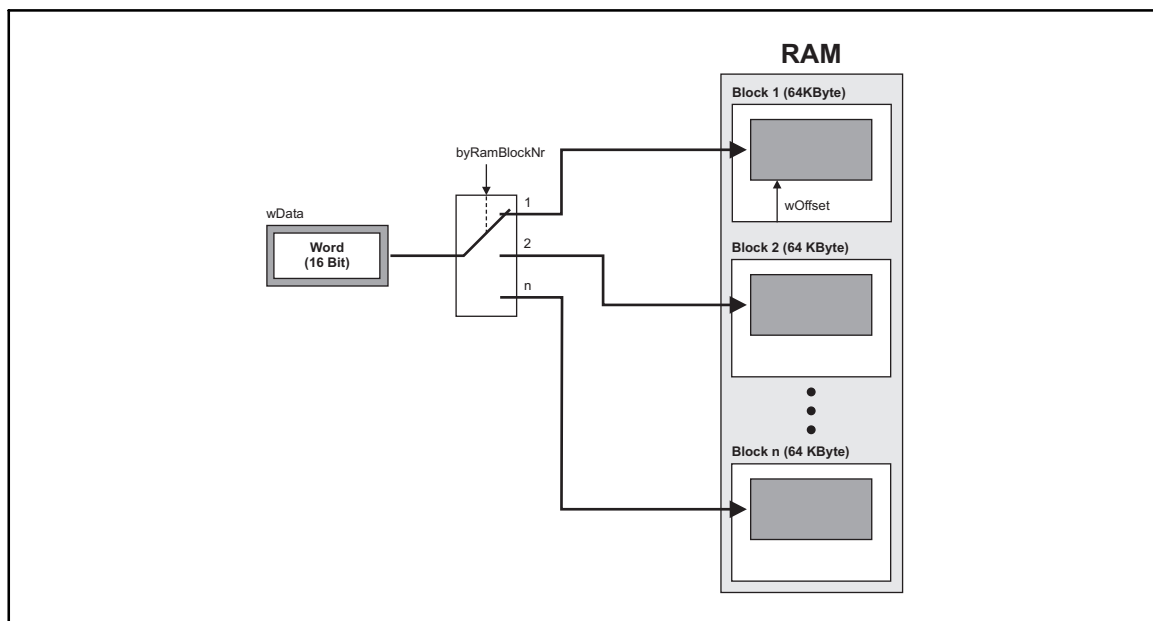
Rückgabewert: Integer	
Wert	Bedeutung
0	Schreibauftrag ist abgeschlossen.
-60	Schreibauftrag wurde nicht an das Betriebssystem übergeben, da Übergabeparameter fehlerhaft: byRamBlockNr ist ungültig.
-35	Kopierauftrag wurde nicht an das Betriebssystem übergeben, da Übergabeparameter fehlerhaft: wOffset ist eine ungerade Adresse oder Wert ist außerhalb des zulässigen Bereichs (0 ... 65534).
-50	Kopierauftrag wurde nicht an das Betriebssystem übergeben, da zur Zeit ein Zugriff auf den Speicher stattfindet (z. B. über Codestellen).



4.11 L_MemWriteWORD

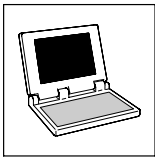
INT **L_MemWriteWORD** (wData, byRamBlockNr, wOffset)

Mit dieser Funktion kann der Wert einer Variable vom Typ "Word" in einen RAM-Block der PLC geschrieben werden.



Übergabeparameter		
Bezeichner	Datentyp	Info/Einstellmöglichkeiten
wData	Word	Zu schreibender Datenwert (16 Bit).
byRamBlockNr	Byte	Auswahl des RAM-Speicherblocks, in den wData geschrieben werden soll.
		1 RAM-Speicher 1, Größe: 64 KByte
		2 RAM-Speicher 2, Größe: 64 KByte
wOffset	Word	Zielposition innerhalb des ausgewählten RAM-Speicherblocks, an die wData geschrieben werden soll.
		0 ... 65534 Zielposition (Offset) • Als Offset sind nur gerade Werte zulässig!

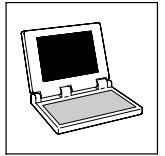
Rückgabewert: Integer	
Wert	Bedeutung
0	Schreibauftrag ist abgeschlossen.
-60	Schreibauftrag wurde nicht an das Betriebssystem übergeben, da Übergabeparameter fehlerhaft: byRamBlockNr ist ungültig.
-35	Kopierauftrag wurde nicht an das Betriebssystem übergeben, da Übergabeparameter fehlerhaft: wOffset ist eine ungerade Adresse oder Wert ist außerhalb des zulässigen Bereichs (0 ... 65534).
-50	Kopierauftrag wurde nicht an das Betriebssystem übergeben, da zur Zeit ein Zugriff auf den Speicher stattfindet (z. B. über Codestellen).



Funktionsbibliothek LenzeMemDrv.lib

Funktionen für RAM-Speicherzugriffe

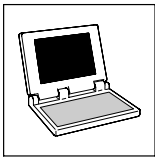
L_MemWriteWORD



5 Funktionen für FLASH-Speicherzugriffe

5.1 Übersicht

	9300 Servo PLC	Drive PLC	ECSxA	
L_MemClearFlash	✓			 5-2
L_MemLoadRamFromFlash	✓			 5-3
L_MemSaveRamToFlash	✓			 5-5



Funktionsbibliothek LenzeMemDrv.lib

Funktionen für FLASH-Speicherzugriffe

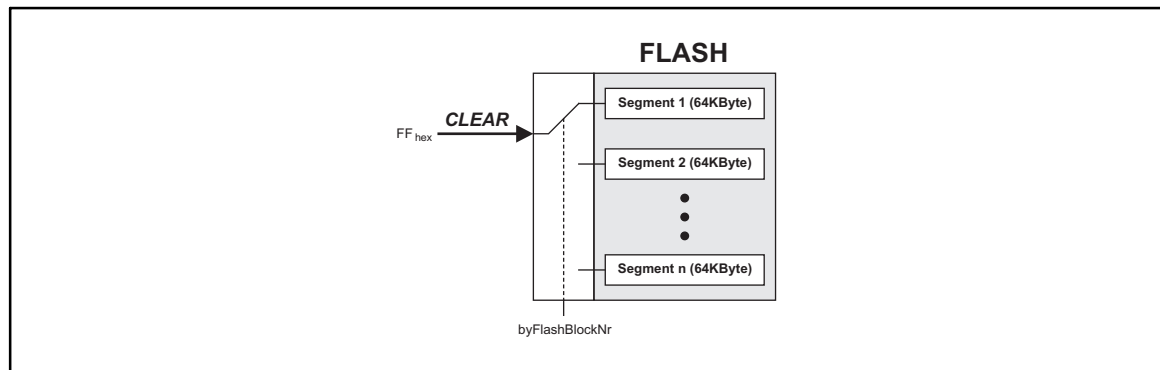
L_MemClearFlash

5.2 L_MemClearFlash

INT **L_MemClearFlash**(byFlashBlockNr)

Mit dieser Funktion kann ein Speichersegment des Flash-Speichers der PLC gelöscht werden.

- Nach dem Löschen des ausgewählten Speichersegments besitzen alle Daten in diesem Segment den Wert "FF_{hex}".
- Die Funktion wird parallel zum PLC-Programm in der Systemtask abgearbeitet, die Bearbeitungsdauer ist daher abhängig von der Auslastung des Systems.
- Der Abschluß des Löschvorgangs kann mit der Funktion **L_MemGetStateDriver** ermittelt werden.



Übergabeparameter		
Bezeichner	Datentyp	Info/Einstellmöglichkeiten
byFlashBlockNr	Byte	Auswahl des zu löschenden Flash-Speichersegments.
		1 Flash-Speichersegment 1, Größe: 64 KByte
		
		15 Flash-Speichersegment 15, Größe: 64 KByte

Rückgabewert: Integer	
Wert	Bedeutung
0	Löschauftrag wurde erfolgreich an das Betriebssystem übergeben.
-40	Kopierauftrag wurde nicht an das Betriebssystem übergeben, da eine zuvor gestartete Speicherverwaltungsfunktion noch den Kopiertreiber belegt.
-50	Kopierauftrag wurde nicht an das Betriebssystem übergeben, da zur Zeit ein Zugriff auf den Speicher stattfindet (z. B. über Codestellen).
-55	Flash-Schreibschutz ist aktiviert; Zugriff verweigert.
-65	Löschauftrag wurde nicht an das Betriebssystem übergeben, da Übergabeparameter fehlerhaft: <i>byFlashBlockNr</i> ist ungültig.
-80	Funktion wird vom Betriebssystem nicht unterstützt.

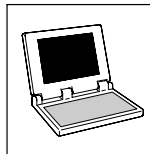
Beispiel

Aufruf der Funktion in ST:

```
(* clear FLASH segment 2 *)

nReturn := L_MemClearFlash(byFlashBlockNr := 2);

(* check function return value *)
```

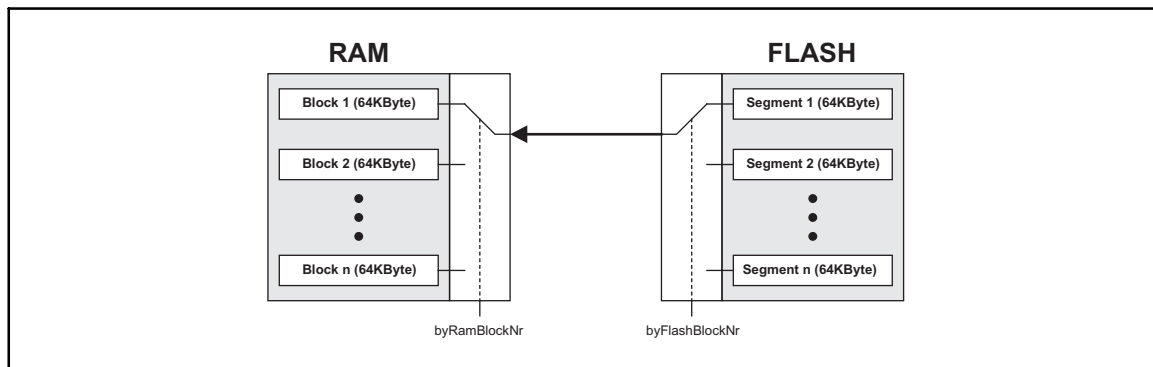


5.3 L_MemLoadRamFromFlash

INT **L_MemLoadRamFromFlash** (byFlashBlockNr, byRamBlockNr)

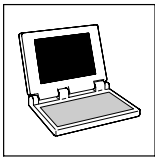
Mit dieser Funktion können die (netzausfallsicheren) Daten eines ganzen Flash-Speichersegments der PLC in einen RAM-Block der PLC geladen werden.

- Die Funktion wird parallel zum PLC-Programm in der Systemtask abgearbeitet, die Bearbeitungsdauer ist daher abhängig von der Auslastung des Systems.
- Der Abschluß des Ladevorgangs kann mit der Funktion **L_MemGetStateDriver** ermittelt werden.



Übergabeparameter		
Bezeichner	Datentyp	Info/Einstellmöglichkeiten
byFlashBlockNr	Byte	Auswahl des Flash-Speichersegments, in dem sich die zu ladenden Daten befinden.
		1 Flash-Speichersegment 1, Größe: 64 KByte
		
		15 Flash-Speichersegment 15, Größe: 64 KByte
byRamBlockNr	Byte	Auswahl des RAM-Speicherblocks, in den die Daten des ausgewählten Flash-Speichersegments geladen werden sollen.
		1 RAM-Speicher 1, Größe: 64 KByte
		2 RAM-Speicher 2, Größe: 64 KByte

Rückgabewert: Integer	
Wert	Bedeutung
0	Kopierauftrag wurde erfolgreich an das Betriebssystem übergeben.
-40	Kopierauftrag wurde nicht an das Betriebssystem übergeben, da eine zuvor gestartete Speicherverwaltungsfunktion noch den Kopiertreiber belegt.
-50	Kopierauftrag wurde nicht an das Betriebssystem übergeben, da zur Zeit ein Zugriff auf den Speicher stattfindet (z. B. über Codestellen).
-60	Kopierauftrag wurde nicht an das Betriebssystem übergeben, da Übergabeparameter fehlerhaft: <i>byRamBlockNr</i> ist ungültig.
-65	Kopierauftrag wurde nicht an das Betriebssystem übergeben, da Übergabeparameter fehlerhaft: <i>byFlashBlockNr</i> ist ungültig.
-80	Funktion wird vom Betriebssystem nicht unterstützt.



Funktionsbibliothek LenzeMemDrv.lib

Funktionen für FLASH-Speicherzugriffe

L_MemLoadRamFromFlash

Beispiel

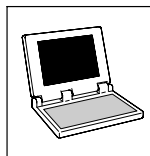
Aufruf der Funktion in ST:

```
(* load from FLASH if last copy action (L_MemLoadRamFromFlash) *)
(* is finished *)

nMemState := L_MemGetStateDriver(byVoid := 0);
bDriverBusy := INT_TO_BOOL(nMemState AND 16#0004);

IF bDriverBusy = FALSE THEN
    nReturn := L_MemLoadRamFromFlash(byFlashBlockNr := 2
                                     byRamBlockNr := 1);
END_IF

(* check function return value *)
```

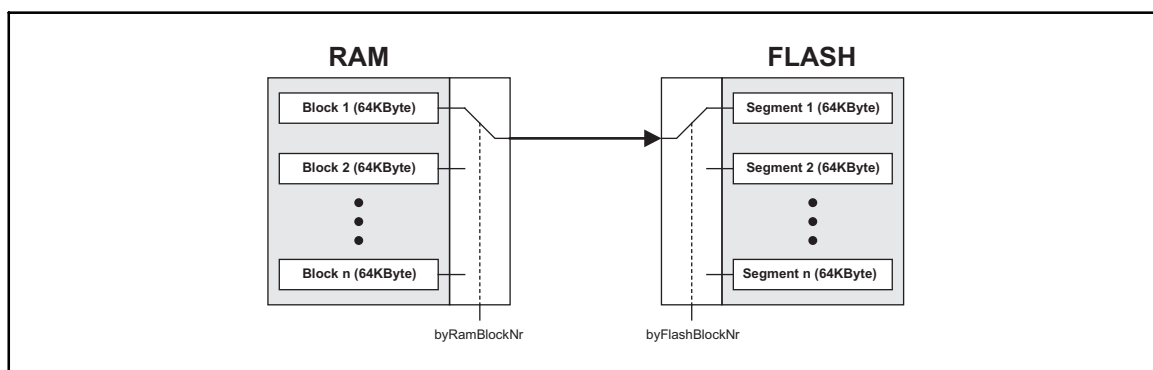


5.4 L_MemSaveRamToFlash

INT **L_MemSaveRamToFlash** (byRamBlockNr, byFlashBlockNr)

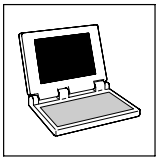
Mit dieser Funktion können die Daten eines ganzen RAM-Blocks der PLC netzausfallsicher in einem Flash-Speichersegment der PLC abgespeichert werden.

- Vor dem eigentlichen Abspeichern der Daten wird von der Funktion zunächst das ausgewählte Flash-Speichersegment gelöscht (mit FF_{hex} überschrieben).
- Die Funktion wird parallel zum PLC-Programm in der Systemtask abgearbeitet, die Bearbeitungsdauer ist daher abhängig von der Auslastung des Systems.
- Der Abschluß des Speichervorgangs kann mit der Funktion **L_MemGetStateDriver** ermittelt werden.



Übergabeparameter		
Bezeichner	Datentyp	Info/Einstellmöglichkeiten
byRamBlockNr	Byte	Auswahl des zu sichernden RAM-Speicherblocks
		1 RAM-Speicher 1, Größe: 64 KByte
		2 RAM-Speicher 2, Größe: 64 KByte
byFlashBlockNr	Byte	Auswahl des Flash-Speichersegments, in das die Daten des ausgewählten RAM-Blocks netzausfallsicher abgespeichert werden sollen.
		1 Flash-Speichersegment 1, Größe: 64 KByte
		...
		15 Flash-Speichersegment 15, Größe: 64 KByte

Rückgabewert: Integer	
Wert	Bedeutung
0	Kopierauftrag wurde erfolgreich an das Betriebssystem übergeben.
-40	Kopierauftrag wurde nicht an das Betriebssystem übergeben, da eine zuvor gestartete Speicherverwaltungsfunktion noch den Kopiertreiber belegt.
-50	Kopierauftrag wurde nicht an das Betriebssystem übergeben, da zur Zeit ein Zugriff auf den Speicher stattfindet (z. B. über Codestellen).
-55	Flash-Schreibschutz ist aktiviert; Zugriff verweigert.
-60	Kopierauftrag wurde nicht an das Betriebssystem übergeben, da Übergabeparameter fehlerhaft: <i>byRamBlockNr</i> ist ungültig.
-65	Kopierauftrag wurde nicht an das Betriebssystem übergeben, da Übergabeparameter fehlerhaft: <i>byFlashBlockNr</i> ist ungültig.
-80	Funktion wird vom Betriebssystem nicht unterstützt.



Funktionsbibliothek LenzeMemDrv.lib

Funktionen für FLASH-Speicherzugriffe

L_MemSaveRamToFlash

Beispiel

Aufruf der Funktion in ST:

```
(* save to FLASH if last copy action (L_MemSaveRamToFlash) *)
(* is finished *)

nMemState := L_MemGetStateDriver(byVoid := 0);
bDriverBusy := INT_TO_BOOL(nMemState AND 16#0008);

IF bDriverBusy = FALSE THEN
    nReturn := L_MemSaveRamToFlash(byRamBlockNr := 1
                                   byFlashBlockNr := 2);
END_IF

(* check function return value *)
```