Ergänzende Information / Additional Information			
Artikel-Nr./Stock No.	X9750375.01	Auftrags-Nr./Order No.	
Produkt/Product	colorSENSOR CFO Option 100	Rahmen-Nr./B-Order No.	
Basis-Betriebsanleitung/Main Manual		colorSENSOR CFO - X9750375	

MICRO-EPSILON

Modbus Dokumentation für colorSENSOR CFO Option 100

1. Einführung

Das Modbus-Protokoll ist ein Single-Master-Protokoll. Der Datenaustausch erfolgt über eine serielle oder über eine Netzwerk (TCP/IP) Schnittstelle. Der Controller funktioniert als Modbus-Slave: Er antwortet auf Anfragen von einem Master.

Das Modbus-Protokoll erlaubt einen teilweisen Zugriff auf die wichtigsten Funktionen des Controllers. Intern verwendet es für alle Operationen die HTTP-basierte API des Controllers.

Die Modbus-Schnittstelle des colorSENSOR unterstützt die folgenden Protokollmerkmale:

- Transport über TCP (IPv4 und IPv6)
- Transport über RS232 und USB mit RTU (Standard) oder ASCII-Format
- Serielle Baudraten: 9600, 19200 (Voreinstellung), 115200

Die Modbus-Slave-Adresse (nur für serielle Verbindungen relevant) ist konfigurierbar. Standardmäßig ist der colorSENSOR CFO nicht an eine bestimmte Adresse gebunden, sondern antwortet auf jedes Paket.

Der vollständige Satz der unterstützten Befehle ist als JSON-Dump verfügbar. Dieser strukturierte Datensatz soll die Erstellung eines herstellerspezifischen Modbus-Mappings für den Controller erleichtern.

2. Schnellstart

Die folgenden Konfigurationsdetails und Hinweise sollen die ersten Schritte mit der Modbus-Protokollimplementierung der Controller erleichtern:

- Schließen Sie den Controller über RS232 (Baudrate: 19200), USB oder TCP (Port 502) an, um über das Modbus-Protokoll zu kommunizieren.
- Verwenden Sie Big-Endian (Byte-Ordnung und Wort-Ordnung) wenn Sie Daten der Modbus-Antworten interpretieren.
- Berücksichtigen Sie das 1-basierte Adressierungsschema beim Zugriff auf Register. Zum Beispiel wird eine dokumentierte Adresse von 501 über die Leitung als 500 übertragen. Die meisten Modbus-Client-Implementierungen wenden diese Übersetzung implizit an. Nur sehr wenige Implementierungen verwenden stattdessen die On-Wire-Adresse. In diesem Fall muss die dokumentierte Adresse für diese spezifischen Clients dekrementiert werden.
- Rufen Sie die Input-Register von 500 bis 508 über eine Modbus-Anfrage ab, siehe 5.1.17. Diese Register enthalten feste Werte in verschiedenen Formaten (z.B. Float, 32 Bit und 64 Bit Integer). Stellen Sie sicher, dass Ihre Client-Implementierung diese Werte entsprechend ihrem dokumentierten Wert korrekt interpretiert (siehe die Dokumentation zum Registerinhalt). Im Falle von Fehlinterpretationen müssen Sie möglicherweise die Bytereihenfolge oder den Adressoffset Ihrer Client-Implementierung anpassen.

3. Unterstützte colorSENSOR-Funktionen

Die Modbus-Schnittstelle der Farbsensoren bietet die meisten Funktionen der folgenden API-Endpunkte:

- */defaults* (only matcher-releated defaults)
- */device*
- */firmware* (only status retrieval; no upgrade)
- */firmware/recovery*
- */firmware/recovery/upgrade-from-current*
- */sensor/samples/current*
- */sensor/matchers*
- */sensor/detectables*
- */sensor/detection-profiles*
- */sensor/detection-profiles/autogain*
- */sensor/detection-profiles/white-reference*
- */sensor/capabilities*
- */system*
- */system/factory-reset*
- */system/reboot*
- */peripherals/outputs*
- */peripherals/rs232*
- */peripherals/usb*
- */settings*

Die folgenden API-Endpunkte werden aufgrund der Unbeständigkeit ihrer Daten oder ihrer Komplexität (die innerhalb der API schwer auszudrücken sind) im Modebus-Protokoll nicht unterstützt.

- */access*
- */action-triggers*
- */actions*
- */firmware/images*
- */firmware/settings*
- */network/interfaces/*
- */peripherals/keypad*
- */peripherals/trigger-sources*
- */system/time*
- */system/time/zones*

4. Datentypen und Registeradressierung

4.1 Datentypen und Modbus-Funktionen

Das Modbus-Protokoll spezifiziert verschiedene Funktionen für den Zugriff auf und die Veränderung von Werten.

Die folgenden Funktionen (und ihre jeweiligen Funktionscodes) werden für die verschiedenen verwendet:

Funktion	Code	Funktionsname
Bits nur lesen	2	Diskrete Eingänge lesen
Beschreibbare Bits	1	Einzelne Bits lesen
	5	Einzelne Bits schreiben
	15	Mehrfachbits schreiben
Wörter nur lesen	4	Eingabe-Register lesen
Wörter schreiben	3	Lesen mehrerer Holding Register
	6	Schreiben eines einzelnen Holding Register
	16	Schreiben mehrerer Holding Register
	23	Lesen/Schreiben mehrerer Register

4.2 Register-Addressen

Die Adressierung von Daten über das Modbus-Protokoll ist nicht streng spezifiziert. Verschiedene Implementierungen verwenden eine Vielzahl von Namensschemata und Offsets. Die relevanten Details dieser Modbus-Implementierung sind:

- Alle in dieser Dokumentation geschriebenen Adressen sind Register-Offsets, die sich auf die spezifische Modbus-Funktion beziehen.
- Alle Adressen sind 1-basiert. Dieser Ansatz wird von den meisten Modbus-Implementierungen verwendet.

Zum Beispiel ist das Register für den Float-Testwert als schreibgeschütztes Wort an Adresse 501 dokumentiert. Diese Adresse könnte auch als 30501 geschrieben werden (basierend auf einem traditionellen Modbus-Adressierungsschema, das die Funktionen auf bestimmte Adressbereiche abbildet). Der Inhalt dieses Registers kann mit der Funktion Read Input Registers (Funktionskennung “4”) abgerufen werden. Die interne Adresse dieses Wertes (wie sie für das On-Wire-Format von Modbus verwendet wird) ist 500 (aufgrund der 1-basierten Registeradressierung). Diese interne Adresse wird nur von sehr wenigen Modbus-Client-Implementierungen verwendet. Die meisten Implementierungen verwenden stattdessen die 1-basierte Adresse.

Clients ohne Unterstützung für Adress-Offsets müssen möglicherweise jede Adresse (wie hier dokumentiert) beim Zusammensetzen des Modbus-Datenrahmens dekrementieren.

4.3 Einfache Datentypen

Die Modbus-Spezifikation beschreibt einfache Datentypen (Bits und 16-Bit-Wörter). Zusätzlich werden die folgenden Datentypen von der Modbus-Implementierung des colorSENSORs verwendet:

- Float-Werte: zwei Register (32 Bit), IEEE-754, Big-Endian-Wortreihenfolge und Byte-Reihenfolge
- Ganzzahlige Werte mit 32 Bit (zwei Register) oder 64 Bit (vier Register): Big-Endian-Wortreihenfolge und Byte-Reihenfolge.
- Zeichenketten: das erste Byte enthält die Länge; alle folgenden Bytes enthalten die ASCII-Zeichen. So enthält das erste „Wort"-Register das Längenbyte und das erste Zeichen. Jedes folgende „Wort"-Register enthält zwei Zeichen. Das Lesen über das Ende der Zeichenkettenlänge hinaus ist erlaubt und gibt Null-Bytes zurück. Daher befindet sich am Ende der Zeichenkette in der Regel ein Null-Byte. Sie dürfen sich jedoch nicht darauf verlassen, da das abschließende Nullbyte fehlt, wenn die Zeichenfolge genau die maximal zulässige Anzahl von Zeichen für diese Zeichenfolge verwendet.
- Bitmaske: 16-Bit-Wörter werden verwendet, um boolesche Felder darzustellen oder zu manipulieren. Jedes Bit repräsentiert einen einzelnen booleschen Wert. Die Beschreibung jedes Bitmasken-Datenfeldes ordnet Bitpositionen dem durch dieses Bit beschriebenen booleschen Zustand zu. Ein Wert von Null wird als falsch (nicht aktiv) angesehen. Ein Wert von Eins ist wahr. Die Bitpositionen beginnen bei Null mit dem niedrigstwertigen Bit.

5. Sitzungsstatus, Aktualität und mehrere Schnittstellen

Mehrere Schnittstellen des Sensors können über das Modbus-Protokoll kommunizieren. Jede Hardwareschnittstelle (z.B. RS232, USB) verwaltet ihren eigenen Zustand. Dies ist relevant für zustandsabhängige Operationen (z.B. Zugriff auf eine Sammlung), die eine Abfolge von Lese- oder Schreibanforderungen erfordern. Die Ethernet-Schnittstelle akzeptiert TCP-Verbindungen. Jede Verbindung verfolgt ihren eigenen Status für die Dauer der Verbindung.

5.1 Funktionen

5.1.1 API-Endpunkt „Einstellen und aktivieren der automatisches Aussteuerung“: /sensor/detection-profiles/current/auto gain

„Autogain-Prozedur“ ausführen, um geeignete Abtasteigenschaften für die aktuelle optische Umgebung zu bestimmen. Die resultierende Abtasteinrichtung wird automatisch angewendet. Diese neuen Einstellungen werden wirksam, sobald die Antwort „erfolgreich abgeschlossen“ gesendet wird. Der Erfolg oder Misserfolg einer Autogain-Prozedur kann verifiziert werden, sobald das „autogain_is_running“-Flag gelöscht wird.

Adresse	Datentyp	Operation	Beschreibung		FC
00020	Bit	schreiben	Autogain starten		5, 15
00302	Bit Maske	lesen	Status des zuletzt gestarteten Autogain-Verfahrens		4
			Position	**Beschreibung**	
			0	läuft noch	
			1	erfolgreich abgeschlossen	
			2	gescheitert: Ziel ist zu dunkel	
00410	Float	lesen / schreiben	Minimal gewünschte Abtastrate		3, 4, 6, 16
00412	Float	lesen / schreiben	Ziel-Analog-Eingangspegel		3, 4, 6, 16
00414	Uint16	lesen / schreiben	Anzahl der für die Mittelwertbildung verwendeten Muster.		3, 4, 6, 16
00415	Bit Maske	lesen / schreiben	Digitale Merker für Autogain-Verfahren Standardwert: 65535		3, 4, 6, 16
			Position	**Beschreibung**	
			0	Internen Emitter aktivieren	
			1	Umgebungslichtkompensation aktivieren	

Adresse	Datentyp	Operation	Beschreibung		FC
00416	Bit Maske	lesen / schreiben	Standard-Autogain-Einstellungen mit benutzerdefinierten Werten überschreiben		3, 4, 6, 16
			Position	**Beschreibung**	
			0	Überschreiben der minimal gewünschten Abtastrate	
			1	Überschreiben des analogen Zieleingang-spegels	
			2	Die Anzahl der für die Mittelwertbildung verwendeten Proben überschreiben	

5.1.2 API-Endpunkt: “Weiß-Referenz setzen / zurücksetzen”: /Sensor/Erkennungsprofile/Strom/Weißreferenz

Die Weiß-Referenz wird zur Berechnung genauer Farbpositionen in den Farbräumen verwendet. Die werkseitig voreingestellte Weiß-Referenz ist für einen bestimmten Satz von Sensor und Optiken geeignet. Eine benutzerdefinierte Weiß-Referenz kann abgetastet werden. Hierfür wird ein Weißstandard empfohlen.

Adresse	Datentyp	Operation	Beschreibung	FC
00021	Bit	schreiben	Weiß-Referenz auf Werkseinstellung zurücksetzen	5
00022	Bit	schreiben	Weiß-Referenz setzen	5

5.1.3 API-Endpunkt „Neue Farbtabelle hinzufügen“: /sensor/matchers

Adresse	Datentyp	Operation	Beschreibung	FC
00024	Bit	schreiben	Erstellen einer neuen Farbgruppe und zuweisen der aktuelle Schaltausgangs (als detektierbar).	5, 15
00451	Uint16	lesen	Abrufen der Kennung der zuletzt erstellten Farbgruppe.	4

5.1.4 API-Endpunkt: „Verwalten der Farben eines Farbgruppen-: /sensor/matchers

Jede Farbgruppe (Matcher) kann sich auf eine oder mehrere Farben (Detectables) beziehen.

Adresse	Datentyp	Operation	Beschreibung	FC
00025	Bit	schreiben	Hinzufügen einer neuen erkennbaren Farbe zu einer vorhandenen Matcher (Farbgruppe).	5
00026	Bit	schreiben	Löschen aller Farben in einer Farbgruppe	5
00027	Bit	lesen	Gibt an, ob die aktuell ausgewählte Farbgruppe existiert.	1
00311	Uint16	lesen	Aktuelle Anzahl der der Farbgruppe zugewiesenen Farbe(Detectables).	4
00450	Uint16	lesen / schreiben	Bestimmen der Farbgruppe zum Hinzufügen oder Entfernen von Farben.	3, 6

5.1.5 API-Endpunkt: „Sensorfähigkeiten lesen”: /sensor/capabilities

Lesen der eingestellten Funktionen im Controller.

Adresse	Datentyp	Operation	Beschreibung		FC
00300	Uint16	lesen	Anzahl der verfügbaren Schaltausgänge		4
00301	Bit Maske	lesen	Vom Sensor unterstützte Farbräume		4
			Position	**Beschreibung**	
			0	XYZ	
			1	L*a*b*	
			2	xyY	
			3	L*u*v*	
			4	L*u’v’	
00303	Bit Maske	lesen	Verfügbare Toleranzformen		4
			Position	**Beschreibung**	
			0	Klassifizierung	
			1	Kugel	
			2	Zylinder	
			3	Box	

Adresse	Datentyp	Operation	Beschreibung		FC
00304	Bit Maske	lesen	Verfügbare Schaltausgangstreiber		4
			Position	**Beschreibung**	
			0	deaktiviert	
			1	NPN	
			2	PNP	
			3	Push-Pull	
00305	Float	lesen	Maximale Abtastrate		4
00307	Uint16	lesen	Maximale Anzahl von Nachweisbarkeiten		4
00308	Uint16	lesen	Maximale Anzahl von Abgleichen		4

5.1.6 API-Endpunkt „Aktuelle Messwerte abrufen“: /sensor/samples/current

Auslesen der neuesten Farbmesswerte. Ein einziger Lesevorgang, der den gesamten Speicherbereich des Musters abdeckt, ist garantiert konsistent. Mehrere Lesevorgänge in Serie werden wahrscheinlich zu einer Kombination von Werten aus den verschiedenen Mustern führen, die in der Zeit zwischen der ersten und der letzten Anfrage gesammelt wurden.

Adresse	Datentyp	Operation	Beschreibung	FC
00150	Uint64	lesen	Zeitstempel des aktuellen Musters	4
00154	Float	lesen	Signal-Level des aktuellen Musters	4
00156	Float	lesen	Darstellung der Farbe im XYZ-Farbraum (Element X)	4
00158	Float	lesen	Darstellung der Farbe im XYZ-Farbraum (Element Y)	4
00160	Float	lesen	Darstellung der Farbe im XYZ-Farbraum (Element Z)	4
00162	Float	lesen	Darstellung der Farbe im aktuell aktiven Farbraum (L*)	4
00164	Float	lesen	Darstellung der Farbe im aktuell aktiven Farbraum (a*)	4
00166	Float	lesen	Darstellung der Farbe im aktuell aktiven Farbraum (b*)	4
00168	Float	lesen	Darstellung der Farbe als RGB-Wert (Wert R)	4
00170	Float	lesen	Darstellung der Farbe als RGB-Wert (Wert G)	4
00172	Float	lesen	Darstellung der Farbe als RGB-Wert (Wert B)	4
00174	Uint16	lesen	Eingänge mit einem High-Level-Ereignis während der letzten Abtastperiode (Bit 0 -> IN0)	4

Adresse	Datentyp	Operation	Beschreibung	FC
00175	Uint16	lesen	Eingänge mit einem Niedrig-Level-Ereignis während der letzten Abtastperiode (Bit 0 -> IN0)	4
00176	Uint16	lesen	Eingänge mit einem Ereignis mit ansteigender Flanke während der letzten Abtastperiode (Bit 0 -> IN0)	4
00177	Uint16	lesen	Eingänge mit einem Ereignis mit fallender Flanke während der letzten Abtastperiode (Bit 0 -> IN0)	4
00178	Uint16	lesen	ID der nächstgelegenen Farbgruppe im Bereich der Farbposition des letzten Musters. Der Wert 65535 wird zurückgegeben, wenn die abgetastete Farbe nicht im Bereich einer der verfügbaren Übereinstimmungen lag.	4
00179	Uint16	lesen	Derzeit aktiver Zustand der Schaltausgänge (Bit 0 -> OUT0)	4
00180	Float	lesen	Abstand (basierend auf den Achsen des aktuell konfigurierten Farbraums) zwischen der zuletzt abgetasteten Farbe und der nächstgelegenen geeigneten Farbgruppe (falls vorhanden). Ein negativer Wert (-1) zeigt an, dass sich keine Farbgruppe im Bereich befindet. 1. Distanzwert	4
00182	Float	lesen	Abstand (basierend auf den Achsen des aktuell konfigurierten Farbraums) zwischen der zuletzt abgetasteten Farbe und der nächstgelegenen geeigneten Farbgruppe (falls vorhanden). Ein negativer Wert (-1) zeigt an, dass sich keine Farbgruppe im Bereich befindet. 2. Distanzwert	4
00184	Float	lesen	Abstand (basierend auf den Achsen des aktuell konfigurierten Farbraums) zwischen der zuletzt abgetasteten Farbe und der nächstgelegenen geeigneten Farbgruppe (falls vorhanden). Ein negativer Wert (-1) zeigt an, dass sich keine Farbgruppe im Bereich befindet. 3. Distanzwert	4

5.1.7 API-Endpunkt „Status der Farbtabellen“: /sensor/detection-profiles/current

Abrufen der aktuellen Verwendung der Farbtabelle

Adresse	Datentyp	Operation	Beschreibung	FC
00309	Uint16	lesen	Aktuelle Anzahl von Übereinstimmungen (Farbgruppen), die in der Farbtabelle gespeichert sind	4
00310	Uint16	lesen	Aktuelle Anzahl der in der Farbtabelle gespeicherten Farben	4

5.1.8 API-Endpunkt „Farbtabellen löschen“: /sensor/matchers

Löschen Sie alle Farben, die in der Farbtabelle gespeichert sind.

Adresse	Datentyp	Operation	Beschreibung	FC
00023	Bit	lesen	Alle gespeicherten Farben entfernen	5, 15

5.1.9 API-Endpunkt „Schaltausgangstreiber einstellen“: /peripherals/outputs

Elektrische Ausgangsleitungen können externe Aktoren in verschiedenen elektrischen Modi ansteuern. Der derzeit aktive Modus kann abgerufen und geändert werden.

Adresse	Datentyp	Operation	Beschreibung		FC
00400	Uint16	lesen / schreiben	Abrufen und Ändern des Stromschaltausgangstreibers. Mögliche Werte:		3, 4, 6, 16
			Beschreibung	**Werte**	
			off	0	
			npn	1	
			pnp	2	
			push-pull	3	

5.1.10 API-Endpunkt „Firmware-Version abfragen“: /firmware

Lesen Sie Informationen über die Firmware.

Adresse	Datentyp	Operation	Beschreibung	FC
00100	Uint16	read	Firmware Version (Major: X.0.0)	4
00101	Uint16	read	Firmware Version (Major: 0.X.0)	
00102	Uint16	read	Firmware Version (Major: 0.0.X)	

5.1.11 API-Endpunkt „Einstellungen zurücksetzen“: /settings

Zurücksetzen der Controllereinstellungen

Adresse	Datentyp	Operation	Beschreibung	FC
00006	Bit	schreiben	Alle Einstellungen zurücksetzen	5

5.1.12 API-Endpunkt „Auf Werkseinstellung zurückstellen“: /system-/factory-reset

Controllerfirmware auf die Werkseinstellungen zurücksetzen und neustarten

Adresse	Datentyp	Operation	Beschreibung	FC
00002	Bit	schreiben	Auslösen eines Werksresets der Firmware und der Einstellungen.	5

5.1.13 API-Endpunkt „System Neustart“: /system/reboot

Neustart aller Controllerkomponenten

Adresse	Datentyp	Operation	Beschreibung	FC
00001	Bit	schreiben	Einen Neustart auslösen	5

5.1.14 API-Endpunkt “Upgrade auf Wiederherstellungs-Firmware”: /system/factory-reset

Gespeichertes Wiederherstellungsabbild durch die aktuelle System-Firmware ersetzen. Dies ist hilfreich, wenn Sie das Wiederherstellungsabbild auf eine neuere Firmware-Version aktualisieren wollen.

Adresse	Datentyp	Operation	Beschreibung	FC
00003	Bit	schreiben	Aktualisieren der Wiederherstellungs-Firmware auf die derzeit laufende Firmware-Version.	5

5.1.15 API-Endpunkt “RS232-Schnittstelle konfigurieren”: /peripherals/rs232

Überprüfen und einstellen der RS232-Schnittstelle des Controllers. Einige Einstellungen beziehen sich auf das Modbus-Slave-Protokoll. Die Modbus-Slave-ID wird für die serielle Kommunikation verwendet, wenn mehr als ein Modbus-Gerät an denselben Bus angeschlossen ist. Das Rahmenformat kann je nach Bedarf des Modbus-Masters geändert werden.

Adresse	Datentyp	Operation	Beschreibung		FC
00430	Uint16	lesen / schreiben	Baudrate der RS232-Schnittstelle		3, 4, 6, 16
			Werte	**Beschreibung**	
			9600	0	
			19200	1	
			115200	2	
00431	Uint16	lesen / schreiben	Baudrate der RS232-Schnittstelle zu verwendendes Protokoll		3, 4, 6, 16
			Werte	**Beschreibung**	
			eliza	0	
			modbus	1	
00432	Uint16	lesen / schreiben	Für das Modbus-Protokoll zu verwendende Slave-ID (1..247)		3, 4, 6, 16
00433	Uint16	lesen / schreiben	Für das Modbus-Protokoll zu verwendendes Rahmenformat mögliche Werte		3, 4, 6, 16
			Werte	**Beschreibung**	
			rtu	0	
			ascii	1	

5.1.16 API-Endpunkt “USB-Schnittstelle konfigurieren”: /peripherals/usb

Überprüfen und einstellen der USB-Schnittstelle des Controllers. Einige Einstellungen beziehen sich auf das Modbus-Slave-Protokoll. Die Modbus-Slave-ID wird für die serielle Kommunikation verwendet, wenn mehr als ein Modbus-Gerät an denselben Bus angeschlossen ist. Das Rahmenformat kann je nach Bedarf des Modbus-Masters geändert werden.

Adresse	Datentyp	Operation	Beschreibung		FC
00440	Uint16	lesen / schreiben	Für die USB-Schnittstelle zu verwendendes Protokoll mögliche Werte		3, 4, 6, 16
			Werte	**Beschreibung**	
			eliza	0	
			modbus	1	
00441	Uint16	lesen / schreiben	Für das Modbus-Protokoll zu verwendende Slave-ID (1..247)		3, 4, 6, 16
00442	Uint16	lesen / schreiben	Für das Modbus-Protokoll zu verwendendes Rahmenformat mögliche Werte		3, 4, 6, 16
			Werte	**Beschreibung**	
			rtu	0	
			ascii	1	

5.1.17 Adressen zum Testen der Datenformate

Einige Register antworten mit festgelegten Festwerten, damit die Richtigkeit des konfigurierten Datenformats leicht überprüft werden kann.

Adresse	Datentyp	Operation	Beschreibung	FC
00500	Uint16	lesen	Ein 16-Bit-Ganzzahlwert, der die Zahl 1234 enthält.	4
00501	Float	lesen	Ein Gleitkommawert, der die Zahl -1.0 enthält.	4
00503	Uint32	lesen	Ein 32-Bit-Ganzzahlwert, der die Zahl 12345678 enthält.	4
00505	Uint64	lesen	Ein 64-Bit-Ganzzahlwert, der die Zahl 123456789012 enthält.	4

MICRO-EPSILON Eltrotec GmbH
Manfred-Wörner-Straße 101 · 73037 Göppingen / Deutschland
Tel. +49 (0) 7161 / 98872-300 · Fax +49 (0) 7161 / 98872-303
info@micro-epsilon.de · www.micro-epsilon.de
Your local contact: www.micro-epsilon.com/contact/worldwide/

X9750375.01-A012090HDR