شبکه MODBUS

مشتمل بر :

محمدرضا ماهر

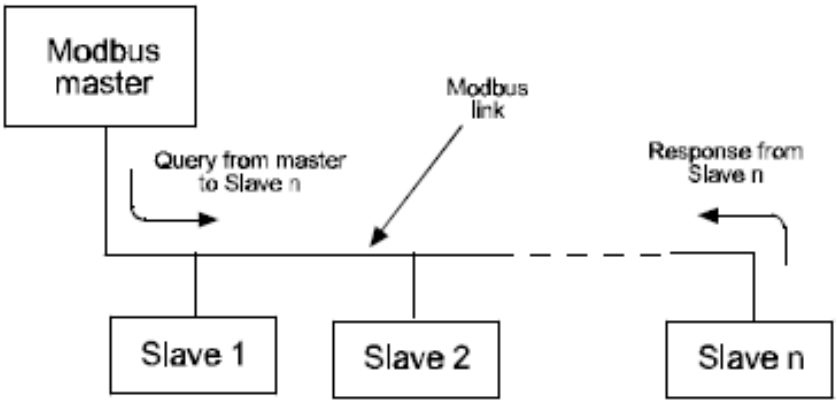
١٣٨٤

٤-١ مقدمه

Modbus یک پروتکل ارتباطی است و ابتدا در سال ۱۹۷۹ توسط Modicon که امروزه Schneider Electric آنرا در بر گرفته عرضه شد. کاربرد اولیه آن برای استفاده در PLC ها بود ولی بتدریج بعنوان یک استاندارد ارتباطی پذیرفته شد و بسیاری از سازندگان تجهیزات اتوماسیون آن را پشتیبانی کردند بدین ترتیب Modbus بصورت یک استاندارد باز در آمد بگونه ای که محصولات سازندگان مختلف بسهولت توسط این پروتکل با یکدیگر ارتباط برقرار کردند. سازندگان وسایل کوچک نیز ترجیح دادند این پروتکل را با ارتباط RS۲۳۲ یا RS۴۸۵ روی وسایل خود بکار ببرند تا استفاده از آنها در پروژه های بزرگ میسر گردد.

Modbus به دلیل استفاده از لینک های سریال RS۲۳۲/RS۴۸۵ دارای محدودیت های شد که به برخی از آنها اشاره میگردد:

- کند بودن خطوط سریال که بین ۹۶۰۰ تا ۱۱۵۰۰۰بیت در ثانیه کار میکنند یعنی در ماکزیمم حالت ۰.۱۱۵ Mbps که این سرعت در مقایسه با شبکه های ارتباطی امروزی که ۱۰۰Mbps یا حتی چند Gbps سرعت دارند پایین است.
- از آنجا که توسط RS۲۳۲ فقط دو وسیله و توسط RS۴۸۵ بین ۲۰ تا ۳۰ وسیله امکان ارتباط دارند از اینرو برای ارتباط دادن تعداد زیادی وسایل مثلاً ۵۰۰ وسیله نیاز به ارتباطات پیچیده درختی شکل است.
- ارتباط سریال Modbus بصورت Master/Slave است بدین معنی که روی باس فقط یک وسیله (Master) اجازه صحبت با گروهی از Slave ها را دارد.



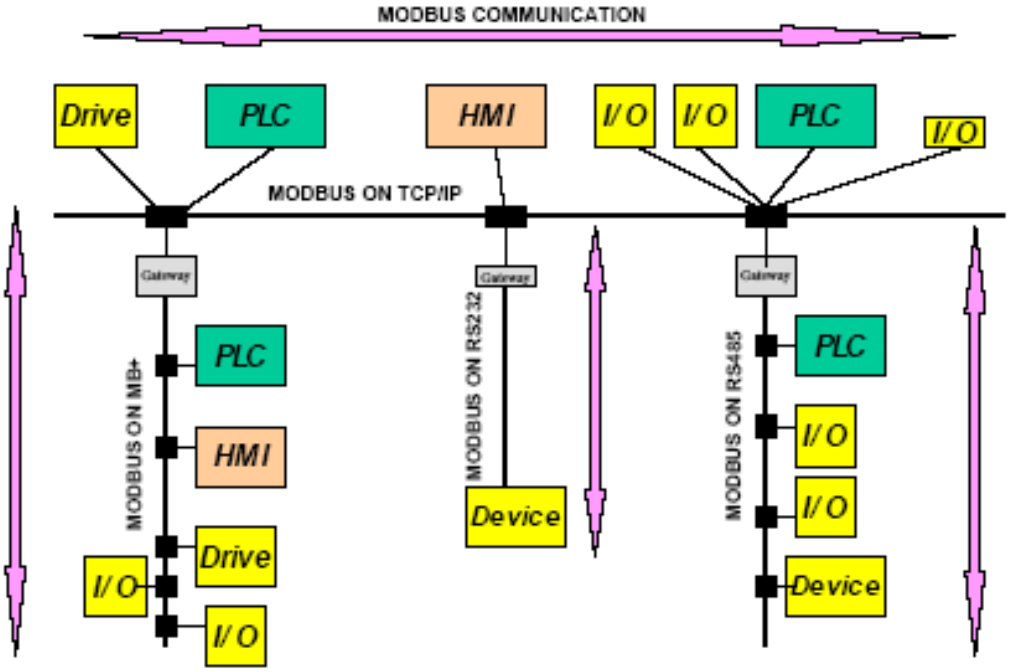
با وجود این محدودیت ها پروتکل Modbus در عرصه اتوماسیون جایگاه ویژه خود را پیدا کرد.

برخی از ویژگیهای پروتکل Modbus نظیر فرمت فریم ها و توالی آنها و فانکشن های کاربردی فیکس هستند برخی دیگر مانند وسیله و مد انتقال توسط کاربر قابل انتخاب هستند.این ویژگیها وقتی سیستم در حال کار است قابل تغییر نمی باشند.

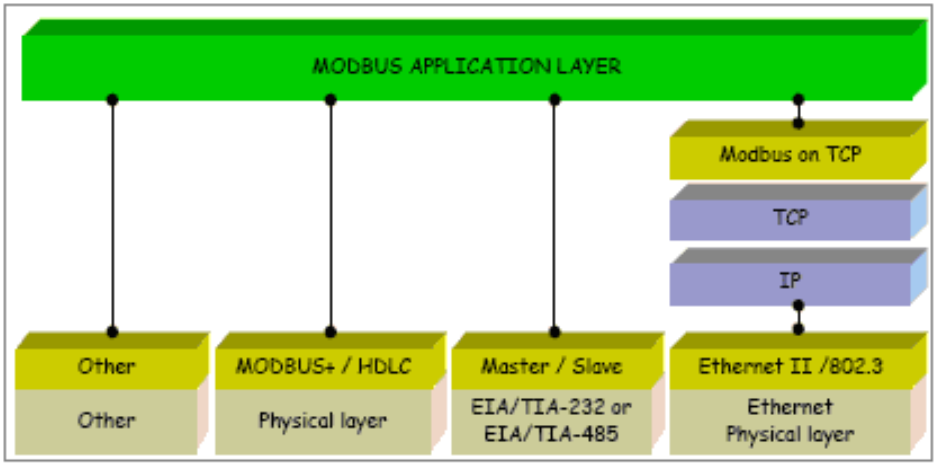
Modbus دارای سه نسخه اصلی زیر است که ویژگی های آنها با هم متفاوتند:

- Modbus RTU/ASCII که بصورت سریال روی RS۴۸۵ یا RS۲۳۲ کار میکند.
- Modbus TCP/IP که روی اترنت کار میکند.
- Modbus Plus که بصورت Token Pass و با سرعت بالا طراحی شده است.

شکل زیر نمونه ای از شبکه Modbus که در آن هر سه نوع پروتکل فوق الذکر بکار رفته اند را نشان میدهد:



شکل زیر مدل OSI را برای انواع پروتکل های Modbus نشان می دهد.



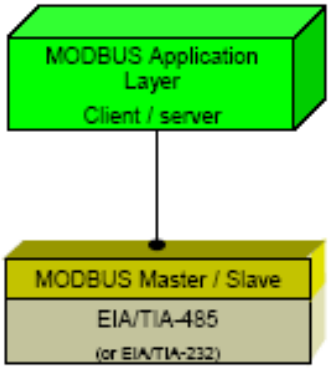
۲-٤ Modbus RTU/ASCII

همانطور که اشاره شد یکی از پروتکل های Modbus برای کاربرد در ارتباط سریال طراحی شده است. این پروتکل که نسخه پایه است از دو مد انتقال ASCII و RTU استفاده می کند

- ASCII قابل خواندن که بعنوان مثال برای تست بکار میرود(فرمت ASCII)
- RTU فشرده و سریع که برای کار نرمال بکار میرود. (فرمت هگزادسیمال)

مد RTU که بعضاً به آن Modbus-B بعنوان Modbus Binary گفته میشود مد اصلی است.مد ASCII که بعضاً به آن Modbus-A نیز گفته میشود برای برخی پیغام ها بکار میرود این پیغام ها طول شان دوبرابر پیغام های RTU می باشد.

Layer	ISO/OSI Model	
7	Application	MODBUS Application Protocol
6	Presentation	Empty
5	Session	Empty
4	Transport	Empty
3	Network	Empty
2	Data Link	MODBUS Serial Line Protocol
1	Physical	EIA/TIA-485 (or EIA/TIA-232)



این پروتکل از لایه های ۱ و۲ و۷ مدل OSI استفاده میکند. که در اینجا به تشریح آنها می پردازیم.

۱-۲-٤ لایه فیزیکی

در لایه فیزیکی Modbus از RS۴۸۵ , RS۲۳۲ استفاده میکند . RS۲۳۲ بصورت نقطه به نقطه P-t-P بسته میشود ولی RS۴۸۵ علاوه بر P-t-P میتواند بصورت Multipoint نیز بسته شود یعنی چندین دستگاه روی یک

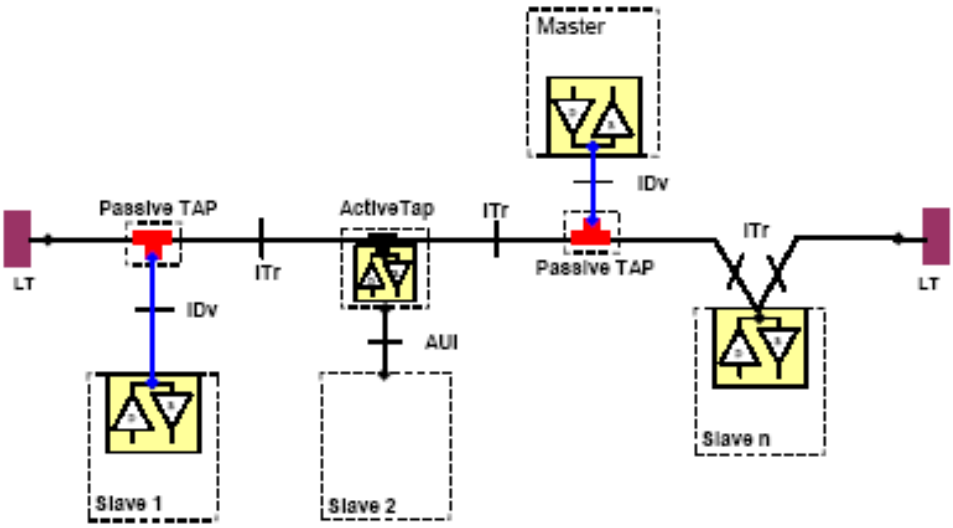
باس قرار گیرند. در اینحالت باس معمولاً ۲ سیمه است اگرچه امکان اتصال ۴ سیمه نیز وجود دارد که تشریح خواهد شد.

٤-۲-۲ اتصال Modbus بصورت RS٤٨٥

در این حالت وسایل بصورت موازی روی کابل Trunk (باس اصلی شبکه) به یکی از سه صورت زیر بسته میشوند:

۱. بصورت Daisy Chain
۲. بصورت Passive Tap و از طریق کابل انشعابی (Derivation Cable) وقتی وسیله مجهز به اینترفیس Modbus باشد
۳. بصورت Active Tap و با کابل AUI وقتی وسیله مجهز به اینترفیس Modbus نباشد

شکل زیر سه نوع ارتباط ذکر شده را نشان میدهد:



در این شکل:

ITr : اینترفیس بین وسیله و کابل Trunk است . Trunk Interface

IDv : اینترفیس بین وسیله Passive Tap است . Derivation Interface

AUI : اینترفیس بین وسیله و Active Tap است . Attachment Unit Interface

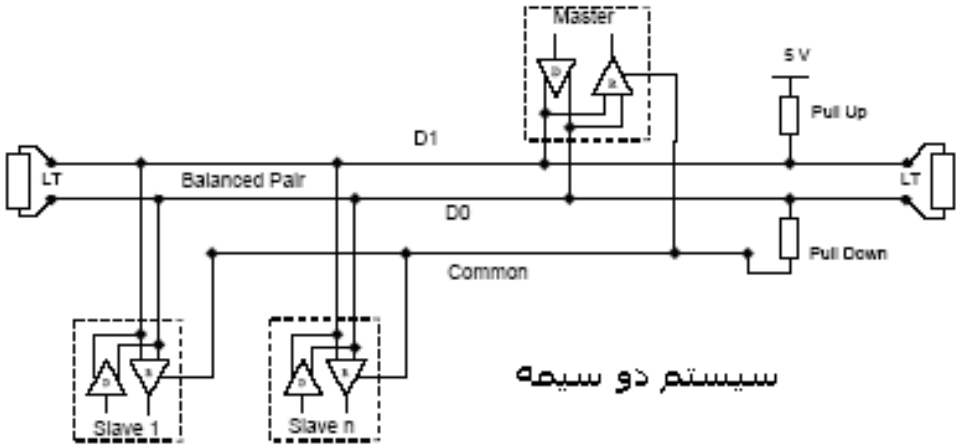
LT : ترمینیتور خط است . Line Termination

باید توجه داشت که:

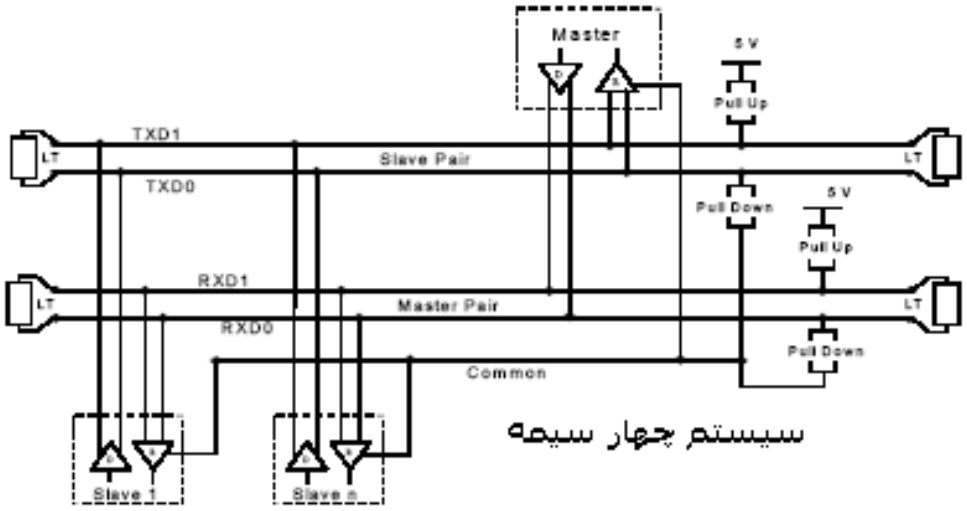
- در برخی موارد ممکن است Tap بصورت مستقیم و بدون کابل انشعابی به سوکت IDv یا سوکت AUI متصل شده باشد.

- یک Tap ممکن است دارای چندین سوکت IDv برای اتصال چند وسیله باشد که دراینحالت به آن Distributor نیز می گویند.
- وقتی از Active Tap استفاده میشود ممکن است تغذیه آن از طریق ITr یا AUI برقرار شود.

اتصال RS۴۸۵ میتواند دوسیمه یا چهار سیمه باشد.(شکل صفحه بعد) اتصال ۴ سیمه اختیاری (optional) است در اینحالت همانطور که در شکل نشان داده شده است از ۵ رشته کابل، ۲ رشته (یک زوج) مربوط به Master است که RXD۱ , RXD۰ نامیده میشود. Slave دیتای Master را از طریق این دو رشته دریافت میکند. پاسخ Slave از طریق دو سیم دیگر TXD۱ و TXD۰ به Master ارسال میگردد. رشته پنجم نیز برای اتصال مشترک (Common) استفاده می گردد.



سیستم دو سیمه



سیستم چهار سیمه

وضعیت خطوط سیگنال برای اتصال ۲ سیمه و ۴ سیمه در جداول بعد آورده شده است:

Required Circuits		For device	Required on device	EIA/TIA-٤٨٥ name	Description
on ITr	on IDv				
D١	D١	I/O	X	B/B'	Transceiver terminal ١, V١ Voltage (V١ > V٠ for binary ١ [OFF] state)
D٠	D٠	I/O	X	A/A'	Transceiver terminal ٠, V٠ Voltage (V٠ > V١ for binary ٠ [ON] state)
Common	Common	--	X	C/C'	Signal and optional Power Supply Common

Required Circuits		For device	Required on device	EIA/TIA-٤٨٥ name	Description for IDv
on ITr	on IDv				
TXD١	TXD١	Out	X	B	Generator terminal ١, Vb Voltage (Vb > Va for binary ١ [OFF] state)
TXD٠	TXD٠	Out	X	A	Generator terminal ٠, Va Voltage (Va > Vb for binary ٠ [ON] state)

RXD۱	RXD۱	In	(۱)	B’	Receiver terminal ۱, Vb’ Voltage (Vb’ > Va’ for binary ۱ [OFF] state)
RXD۰	RXD۰	In	(۱)	A’	Receiver terminal ۰, Va’ Voltage (Va’ > Vb’ for binary ۰ [ON] state)
Common	Common	--	X	C/C’	Signal and optional Power Supply Common

در اتصال ۴ سیمه کابل انشعاب بصورت Cross متصل میشود (مطابق جدول زیر). بهترین روش اتصال در ۴ سیمه استفاده از Tap هایی است که روی خودشان اتصالات را Cross کرده اند.

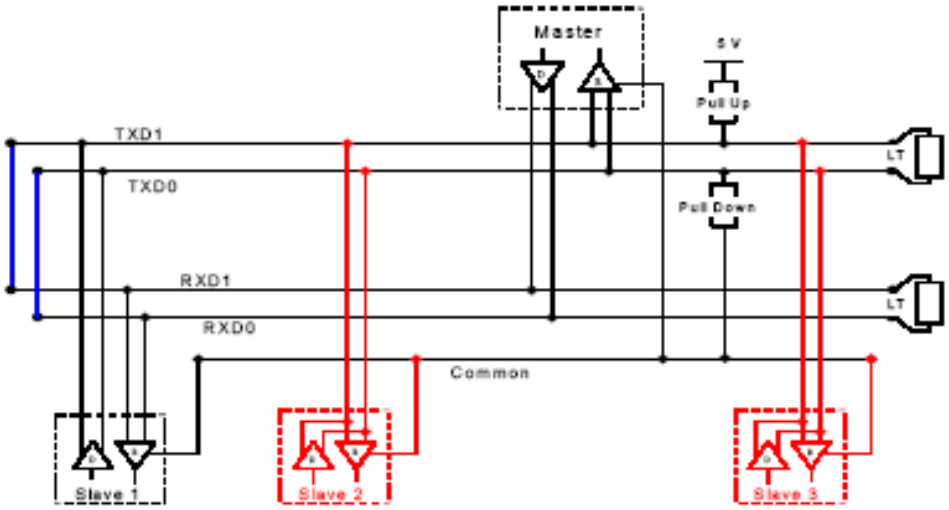
در اتصال ۲ سیمه کابل انشعابی نباید Cross بسته شود.

	Signal on Master IDv		**EIA/TIA–٤٨٥ Name**	**Circuit on ITr**
	Name	**Type**		
Slave Pair	RXD۱	In	B’	TXD۱
	RXD۰	In	A’	TXD۰
Master Pair	TXD۱	Out	B	RXD۱
	TXD۰	Out	A	RXD۰
	Common	--	C/C’	Common

بین سیستم ۴ سیمه و ۲ سیمه می توان سازگاری برقرار کرد. دو حالت زیر را در نظر بگیرید:

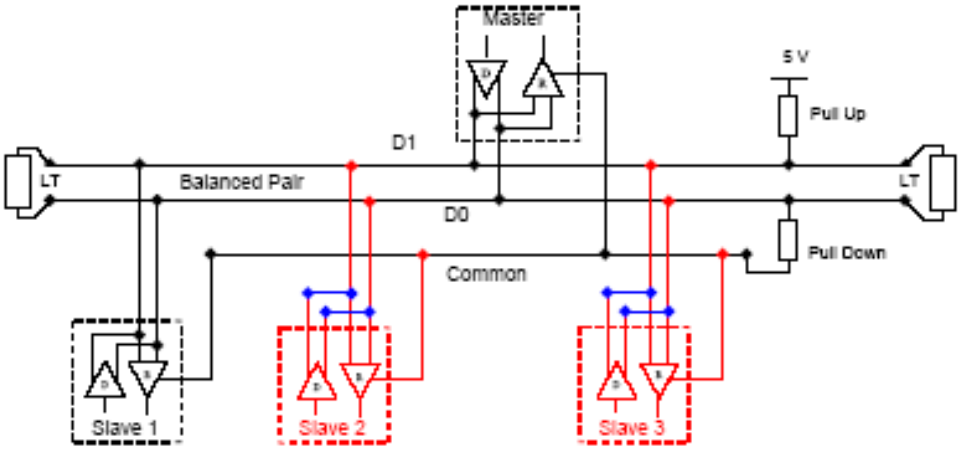
حالت اول : اگر سیستم ۴ سیمه ای از قبل موجود باشد و بخواهیم از آن بعنوان ۲ سیمه استفاده کنیم لازم است

- سیم های سیگنال TXD۰ و RXD۰ را به هم ببندیم تا خط D۰ ایجاد شود .
- سیم های سیگنال TXD۱ و RXD۱ را به هم ببندیم تا خط D۱ ایجاد شود .
- اتصال ترمینیتور ها را از نو آرایش دهیم.



حالت دوم : اگر سیستم ۲ سیمه ای از قبل موجود باشد و بخواهیم وسیله ای ۴ سیمه را به آن متصل کنیم لازم است روی وسیله :

- سیم های سیگنال TXD۰ و RXD۰ را به سیم D۰ ببندیم.
- سیم های سیگنال TXD۱ و RXD۱ را به سیم D۱ ببندیم.

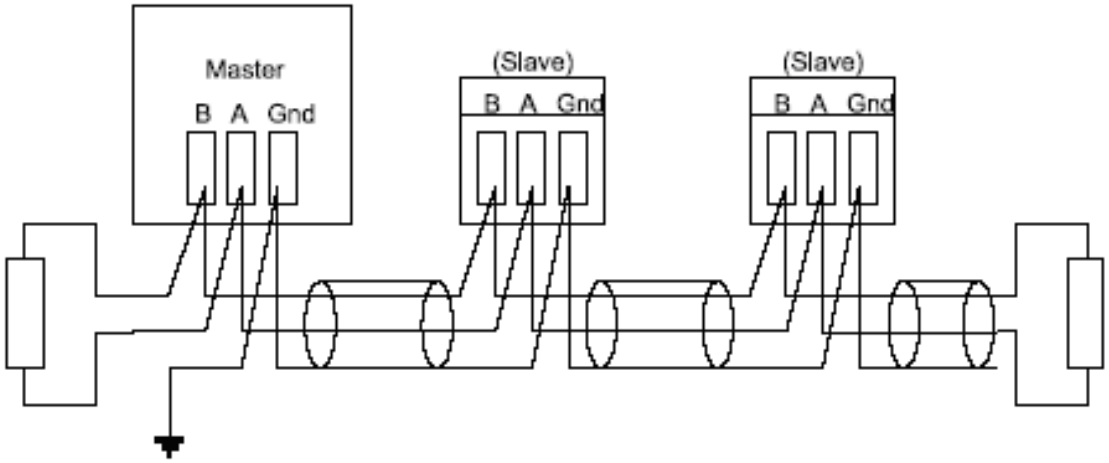


در اتصال RS۴۸۵ چه بصورت ۲ سیمه و چه بصورت ۴ سیمه در نظر داشتن نکات زیر ضروری است:

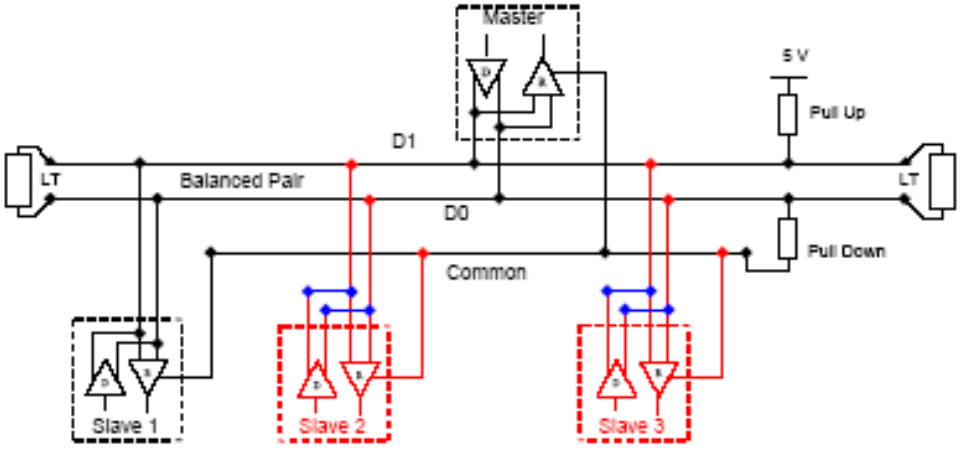
- بطور معمول بدون ریپیتر ماکزیمم ۳۲ وسیله را میتوان روی Modbus RS۴۸۵ متصل نمود.
- توپولوژی اتصال بصورت Bus است که میتواند Daisy Chain یا Tap and Drop باشد.
- طول کابل Trunk محدود است این محدودیت بستگی به سرعت انتقال دیتا ، مشخصات کابل(از جمله خازن کابل) ، تعداد Node های متصل شده بصورت Daisy Chain و نیز ۲سیمه یا ۴ سیمه

بودن سیستم دارد. برای سرعت bps ۹۶۰۰ با کابل AWG۲۶ یا ضخیم تر و سیستم ۲ سیمه این طول ۱۰۰۰ متر است . برای سیستم ۴ سیمه با مشخصات فوق این طول به نصف کاهش می یابد

- طول کابل انشعابی کوتاه باشد و از ۲۰ متر تجاوز نکند. کلاً اگر n انشعاب وجود داشته باشد برای بدست آوردن ماکزیمم طول هر انشعاب عدد ۴۰ متر را بر n تقسیم میکنیم.
- سیم مشترک(GND یاCommon) به زمین (PE) متصل شود ترجیحاً در یکطرف و آنهم در سمت Master



- دوطرف کابل Trunk یعنی باس اصلی توسط ترمینیتور بسته شود. هر زوج سیم دیتا در دوطرف ترمینیتور لازم دارد پس در سیستم ۴ سیمه در هر طرف ۲ ترمینیتور لازم است.
- هیچگاه از ترمینیتور روی کابل انشعابی استفاده نکنید.
- ترمینیتور میتواند یک مقاومت ۱۵۰ اهمی باشد. در برخی کاربردها از مقاومت ۱۲۰ اهمی که با خازن (۱nF , ۱۰V minimum) سری شده استفاده می گردد.
- برای پلاریزه کردن خط از یک جفت مقاومت Pull up و Pull Down که مقدار آنها میتواند بین ۴۵۰ تا ۶۵۰ اهم باشد در یکطرف استفاده میگردد. منظور از پلاریزه کردن خط مصون نگه داشتن باس از تداخل و نویز در حالتی است که روی خط هیچ دیتایی تبادل نمیشود لازم است در اینحالت گیرنده حالت ثابت خود را حفظ کند.
- مقاومت Pull up روی خط D۱ و به ولتاژ ۵ ولت متصل میگردد.
- مقاومت Pull Down روی خط D۰ و به سیم مشترک (Common) متصل میگردد.
- پلاریزاسیون خط در یکطرف و ترجیحاً در سمت Master انجام میشود.
- ماکزیمم وسایل مجاز روی Modbus با پلاریزاسیون به یک چهارم حالت نرمال کاهش می یابد.



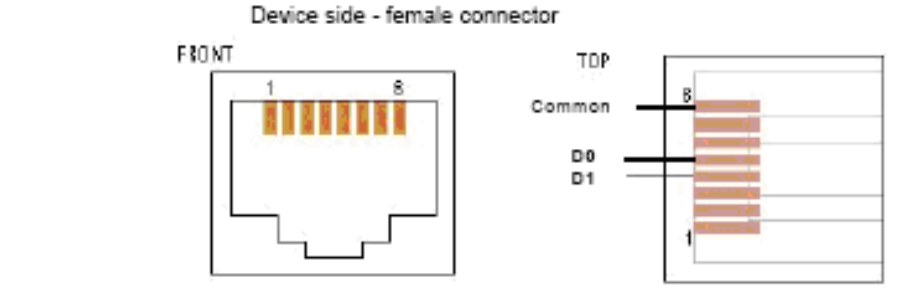
٤-٢-٣ اتصال Modbus بصورت RS۲۳۲

- این ارتباط بصورت نقطه به نقطه است یعنی فقط دو وسیله میتوانند با هم ارتباط داشته باشند .
- برای مسافت های کوتاه (معمولاً) کمتر از ۲۰ متر بکار می رود.
- ترمینیتور ندارد.
- اتصال بین دو وسیله DCE و DTE روی Modbus RS۲۳۲ بصورت جدول زیر است . در این جدول مواردی که با X مشخص شده برای ایجاد ارتباط کافی است .
- سر سیم TXD از یک سمت با سر سیم RXD طرف مقابل مرتبط میشود.
- ممکن است RTS به CTS سمت مقابل و DTR به DSR سمت مقابل وصل شود.

Signal	For DCE	Required on DCE (۱)	Required on DTE (۱)	Description
Common	--	X	X	Signal Common
CTS	In			Clear to Send
DCD	--			Data Carrier Detected (from DCE to DTE)
DSR	In			Data Set Ready
DTR	Out			Data Terminal Ready
RTS	Out			Request to Send
RXD	In	X	X	Received Data
TXD	Out	X	X	Transmitted Data

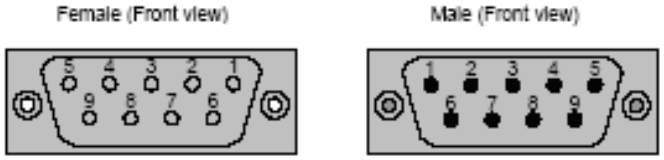
اتصالات و کانکتور ها

کانکتور می تواند RJ۴۵ یا D-shell-۹ pin باشد. در شکل زیر این کانکتورها برای سیستم ۲ سیمه نشان داده شده اند.



گانگتورهاي
سيستم 2 سيمه

2W- MODBUS on RJ45 connector (required pin-out)



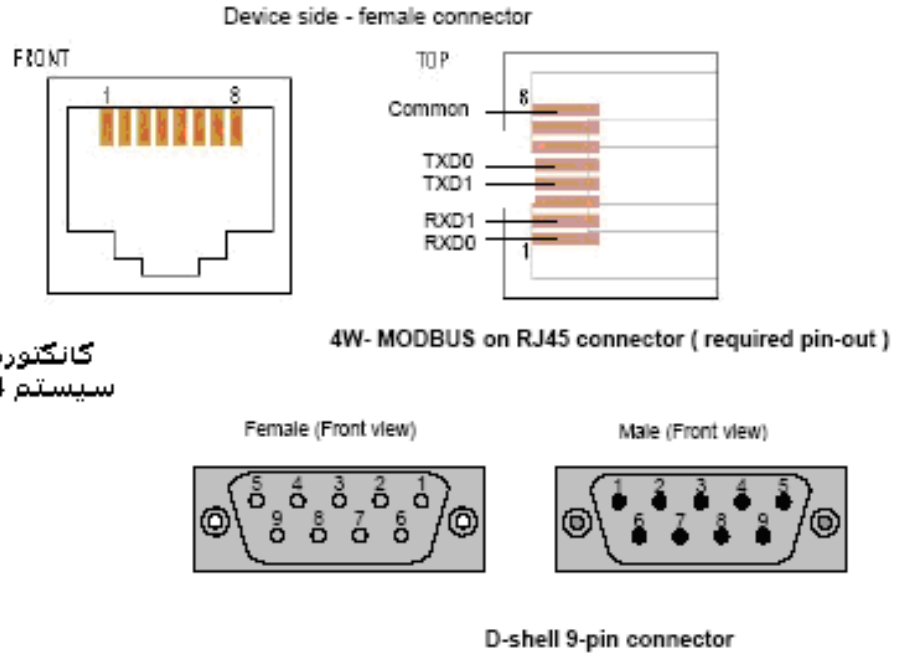
D-shell 9-pin connector

پین های مربوطه برای کانکتورهای فوق در جدول زیر مشخص گردیده اند.

Pin on RJ٤٥	Pin on D٩-shell	Level of requirement	IDv Circuit	ITr Circuit	EIA/TIA–٤٨٥ name	Description for IDv
٣	٣	optional	PMC	--	--	Port Mode Control
۴	۵	required	D١	D١	B/B’	Transceiver terminal ١, V١ Voltage (V١ > V٠ for binary ١ [OFF] state)
۵	٩	required	D٠	D٠	A/A’	Transceiver terminal ٠, V٠ Voltage (V٠ > V١ for binary ٠ [ON] state)
٧	٢	recommended	VP	--	--	Positive ۵...٢۴ V D.C. Power Supply
٨	١	required	Common	Common	C/C’	Signal and Power Supply Common

کانکتورها و مشخصات پین ها برای سیستم ۴ سیمه در صفحه بعد آورده شده است.

پیشنهاد شده که در کابل Modbus سیستم ۲ سیمه، سیم D٠ به رنگ قهوه ای ، سیم D١ به رنگ زرد باشد در سیستم ۴ سیمه ایندو با همین رنگها برای TXD٠ و TXD١ بکار رفته و سیم RXD٠ به رنگ سفید و سیم RXD١ به رنگ آبی باشد . سیم مشترک در هر دو سیستم خاکستری رنگ است. استفاده از این رنگ ها الزامی نیست کما اینکه در استفاده از کابل Cate ۵ رنگ بندی متفاوت خواهد بود.



گانگنورهاي
سيستم 4 سيمه

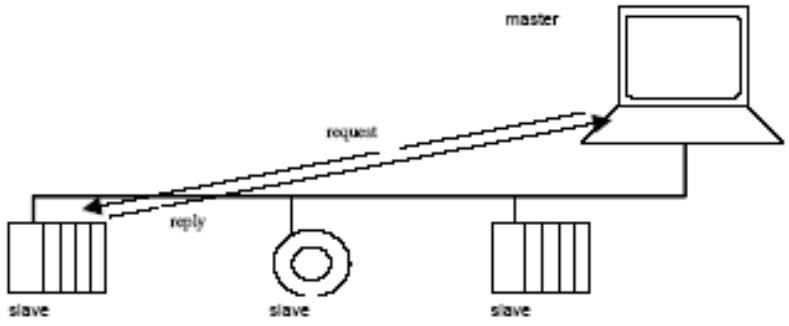
Pin on RJ٤٥	Pin on D٩-shell	Level of requirement	IDv Signal	ITr Signal	EIA/TIA-٤٨٥ name	Description for IDv
١	٨	required	RXD٠	RXD٠	A’	Receiver terminal ٠, Va’ Voltage (Va’ > Vb’ for binary ٠ [ON] state)
٢	٤	required	RXD١	RXD١	B’	Receiver terminal ١, Vb’ Voltage (Vb’ > Va’ for binary ١ [OFF] state)
٣	٣	optional	PMC	--	--	Port Mode Control
٤	٥	required	TXD١	TXD١	B	Generator terminal ١, Vb Voltage (Vb > Va for binary ١ [OFF] state)
٥	٩	required	TXD٠	TXD٠	A	Generator terminal ٠, Va Voltage (Va > Vb for binary ٠ [ON] state)
٧	٢	recommended	VP	--	--	Positive ٥...٢٤ V DC Power Supply
٨	١	required	Common	Common	C/C’	Signal and Power Supply Common

٤-۲-٤ لایه دیتا لینک (لایه دوم)

این لایه از تکنیک دسترسی Master/Slave استفاده میکند. بنابراین در هر لحظه فقط یک Master میتواند باس را در اختیار داشته و با ماکزیمم ۲۴۷ وسیله Slave که به همان باس متصل هستند ارتباط برقرار کند. بدیهی است در این تکنیک دسترسی Slave ها هیچگاه بدون درخواست Master دیتایی را نمیفرستند و هیچگاه نیز با یکدیگر ارتباط برقرار نمی کنند.

Master به یکی از دو روش زیر درخواست خود را ارسال می نماید:

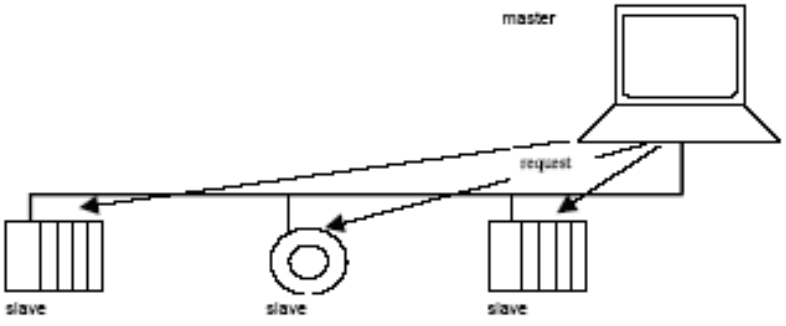
۱. مد Unicast : در این حالت Master از Slave خاصی درخواست دیتا می نماید. Slave پس از دریافت Request پیام reply را به Master می فرستد. بدیهی است هر Slave باید دارای آدرس خاص و منحصر بفردی باشد تا Master بتواند با آن ارتباط برقرار کند.



Unicast mode

۲. مد Broadcast : در این حالت Master پیام خود را به تمام Slave ها میفرستد ولی هیچ پاسخی به Master بر نمیگردد. این مد از جمله برای نوشتن فرامین (Writing Commands) بکار میرود.

برای مد Broadcast آدرس صفر رزرو شده است.

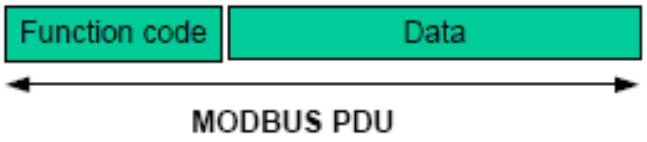


Broadcast mode

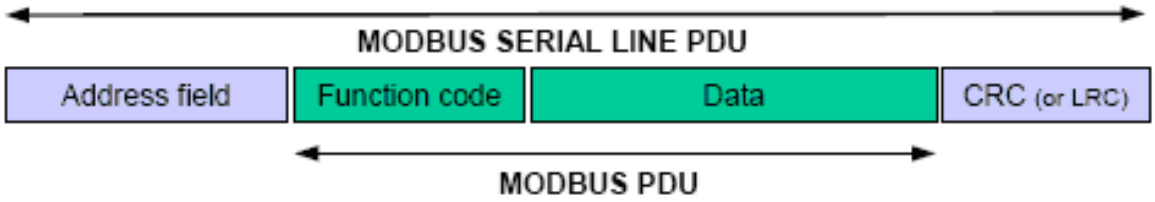
بطور کلی روش آدرس دهی در Modbus مانند جدول بعد است . باید توجه داشت که Master نیاز به هیچ آدرس خاصی ندارد و صرفاً Slave ها هستند که نیاز به آدرس دارند.

فریم اطلاعات

پروتکل Modbus دارای PDU(Protocol Data Unint) های ساده ای مانند شکل زیر است.



دیتاهای دیگری نیز با PDU جمع شده و فریم دیتا را می سازند. این فریم در سمت Master ساخته میشود



فیلد های فریم فوق بشرح زیر هستند:

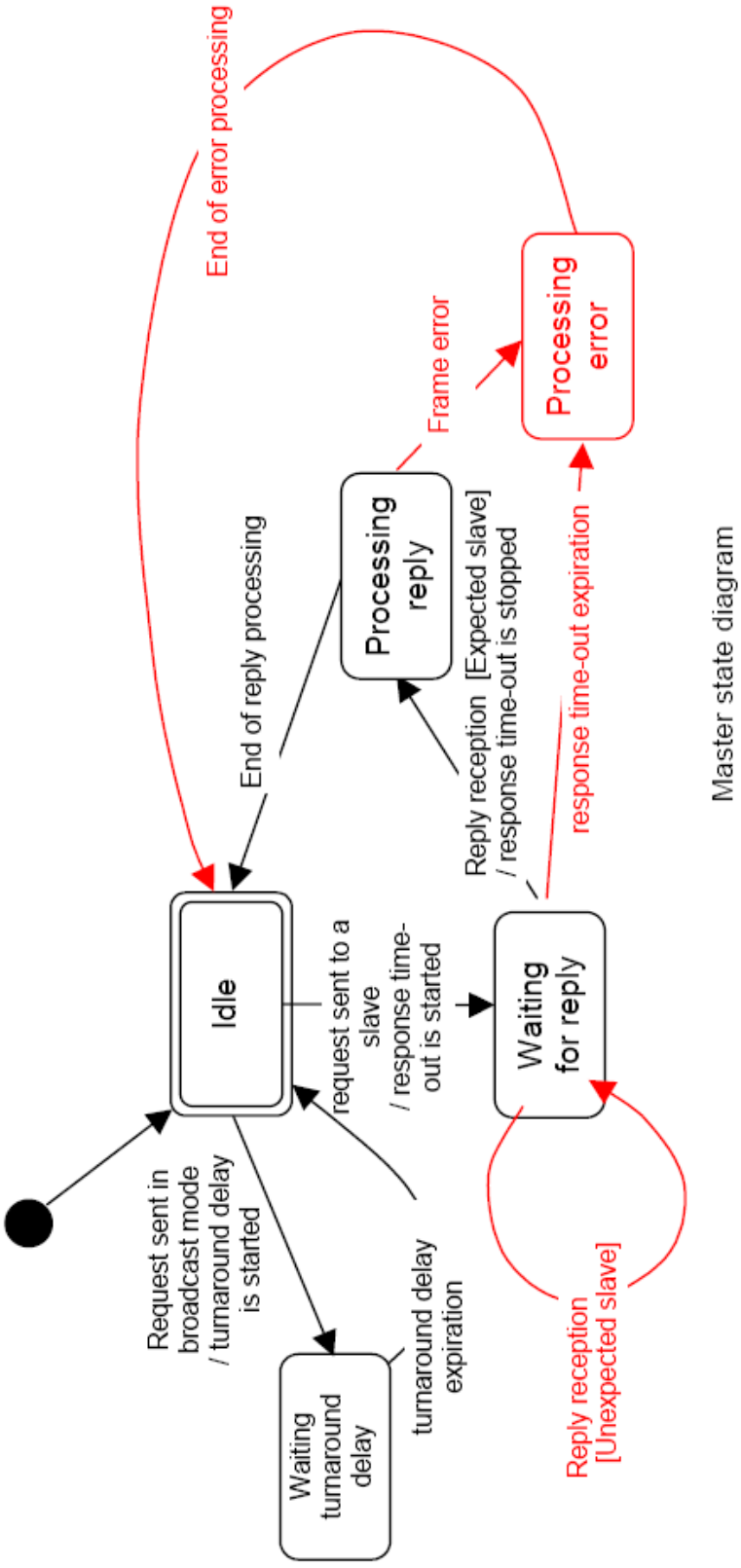
- فیلد آدرس صرفاً حاوی آدرس Slave است و میتواند بین ۱ تا ۲۴۷ باشد . Master در ارتباط Unicast آدرس را در این فیلد قرار می دهد و Slave نیز وقتی پاسخ میدهد آدرس خودش را در این فیلد می گذارد تا Master بفهمد کدام Slave پاسخ داده است.
- فیلد Function Code نشان دهنده عملی است که باید انجام شود (Action)
- فیلد CRC برای Error Checking است که تشریح خواهد شد.

دیاگرام وضعیت Master/Slave

لایه دیتا لینک دارای دو زیر لایه است یکی پروتکل Master/Slave و دیگری مد انتقال که میتواند ASCII یا RTU باشد. دیاگرام کلی وضعیت Master/Slave بصورت خیلی ساده در شکل صفحه بعد آورده شده و در ارتباط با آن نکات زیر حائز اهمیت است:

- وضعیت Idle که به معنای بیکار بودن است وضعیت اولیه است که پس از وصل تغذیه بوجود می آید.در این وضعیت Master پیامی را بصورت Broadcast ارسال می دارد سپس وضعیت Idle را ترک می کند.

- وقتی درخواستی بصورت Unicast به Slave خاصی ارسال شد ، Master منتظر جواب می ماند اما نه بطور نامحدود بلکه با Time out چک میشود.

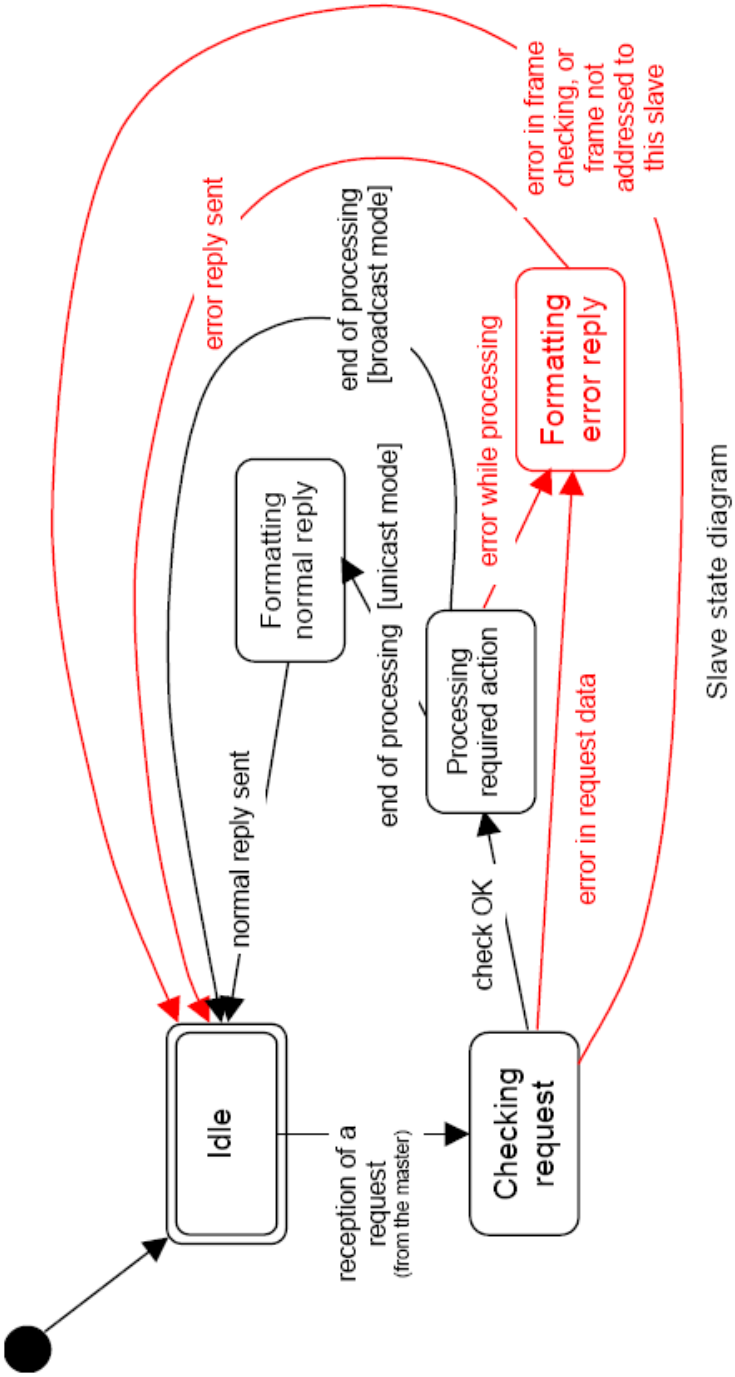


Master state diagram

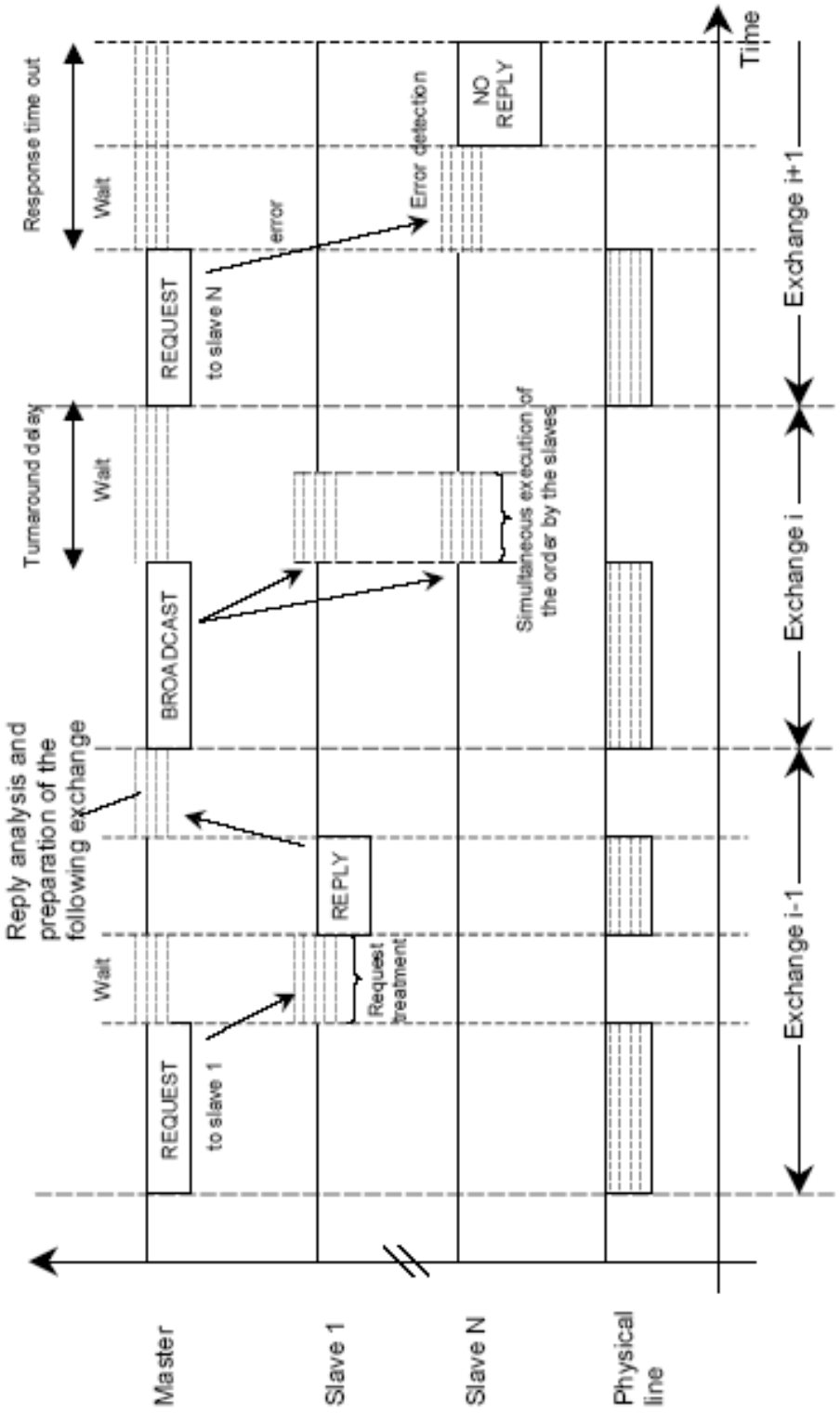
- وقتی پاسخ دریافت شد Master قبل از پردازش دیتا ، پاسخ را چک میکند ممکن است اشکالی وجود داشته باشد مثلاً Slave دیگری پاسخ داده باشد یا در فریم دیتای دریافت شده خطایی رخ داده باشد دراینجا نیز Time out چک میشود تا بلکه در زمان تعیین شده پاسخ صحیح از Slave برسد. اگر پاسخی دریافت نشد و Time out به سر رسید سیگنال Error تولید میشود و Master به وضعیت Idle میرود و تلاش دوباره بسته به تنظیمات انجام شده روی Retry انجام میشود.
- وقتی درخواستی بصورت Broadcast روی باس سریال ارسال می شود پاسخی از طرف Slave ها داده نشده با این وجود Master تاخیری را برای ارسال پیام مورد نظر بعدی در نظر می گیرد تا مطمئن شود Slave ها پیام اولیه را گرفته اند به این تاخیر Turnaround Delay میگویند . Master قبل از برگشت به وضعیت Idle و آمادگی برای ارسال پیام بعدی لحظاتی را بصورت انتظار میگذراند که در دیاگرام فوق بعنوان Waiting Turnaround Delay آمده است این زمان لازم است به حد کافی طولانی باشد تا Slave ها فرمان را پردازش کرده و برای گرفتن دیتای بعدی آماده شوند.
- در حالت Unicast زمان Time out بایستی در حد مناسب و کافی تنظیم شده باشد تا Slaveبتواند در طول آن مدت درخواست را پردازش کرده و جواب را ارائه دهد معمولاً این زمان که به آن Response Time میگویند برای سرعت bps ۹۶۰۰ بین یک تا چند ثانیه تنظیم می شود.
- زمان Turnaround Delay بایستی کمتر از زمان Response Time باشد. این تاخیر برای سرعت bps۹۶۰۰ حدود ۱۰۰ تا ۲۰۰ میلی ثانیه است.
- برای تامین امنیت فریم از دو تکنیک استفاده میشود اول تکنیک Parity که براساس بیتهای یک دیتا مقدار می گیرد دوم چک Redundancy برای کل دیتای داخل فریم . این موارد بعداً تشریح خواهند شد.

دیاگرام وضعیت Slave

رفتار Slave در دیاگرام زیر آمده است همانطور که ملاحظه میشود در اینجا نیز وضعیت Idle وضعیت اولیه پس از وصل تغذیه است. وقتی درخواست دریافت شد Slave بسته دیتا را قبل از اینکه Actionمورد درخواست در آن را انجام دهد چک میکند. در صورتی که اشکالاتی مانند اشکال در فرمت فریم درخواست یا Action ناشناخته و امثال آن وجود داشته باشد پاسخی که نشان دهنده وجود اشکال است به Master اعلام میگردد. در غیر اینصورت Action را انجام داده و سپس پیام مورد نیاز را به Master بر می گرداند.



دیاگرام زمانی ارتباط Master /Slave که سه سناریوی مختلف را نشان میدهد در صفحه بعد آمده است. توجه داشته باشید که طول مدت فازهای Request و Reply و Broadcast بستگی به مشخصات ارتباطی از جمله طول فریم دارد . همچنین طول فازهای Wait و Treatment بستگی به زمان پردازش درخواست توسط Slave دارد.


Time
Response time out
Wait
Error detection
NO REPLY
error
REQUEST
to slave N
Exchange i+1
Turnaround delay
Wait
Simultaneous execution of the order by the slaves
BROADCAST
Exchange i
Reply analysis and preparation of the following exchange
REPLY
Request treatment
Wait
REQUEST
to slave 1
Exchange i-1
Master
Slave 1
Slave N
Physical line

مد های انتقال سریال

دو مد در انتقال سریال Modbus وجود دارد که عبارتند از : RTU و ASCII. این مد ها وضعیت بیت های پیام و نحوه بسته بندی و باز شدن بسته پیام را مشخص میکند. تمام وسایل باید بتوانند مد RTU را که مد پیش فرض است بکار ببرند ولی مد ASCII اختیاری است.

مد RTU یا Remote Terminal Unit

در این مد هر ۸ بیت از بایت پیام شامل دو کاراکتر ۴ بیتی هگزا دسیمال است. این ویژگی چگالی دیتا را افزایش داده و باعث می شود که نسبت به مد ASCII نرخ تبادل دیتا بهتر باشد. هر پیام بصورت رشته ای از کاراکترها ارسال میگردد . فرمت ۱۱ بیت بسته دیتا در مد RTU بصورت زیر است:

- ۱ بیت برای شروع دیتا
- ۸ بیت برای دیتا
- ۱ بیت برای Parity
- ۱ بیت برای پایان دیتا

Start	۱	۲	۳	۴	۵	۶	۷	۸	Par	Stop

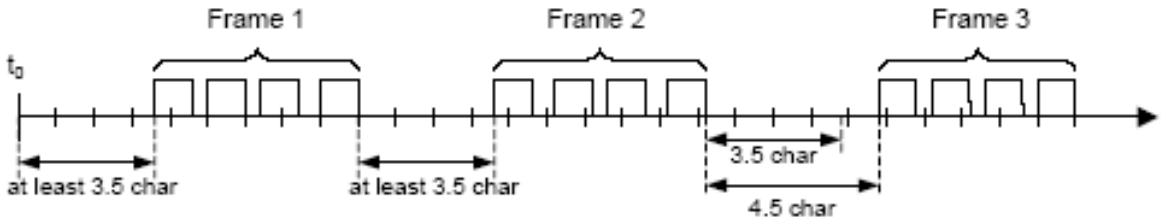
معمولاً Parity زوج بصورت پیش فرض مورد استفاده قرار میگیرد اگرچه Parity فرد یا حتی بدون چک Parity نیز میتواند بکار رود. حالت اخیر برای اطمینان از ماکزیمم سازگاری با محصولات مختلف پیشنهاد میشود در این حالت ۲ بیت پایانی مورد نیاز است شکل زیر:

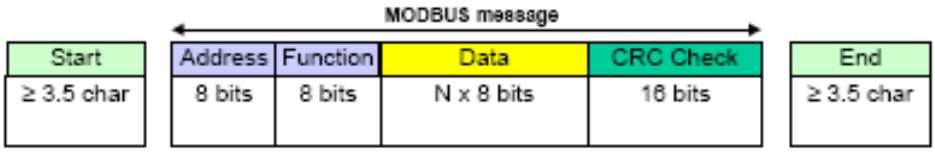
Start	۱	۲	۳	۴	۵	۶	۷	۸	Stop	Stop

فرمت کلی فریم که شامل کاراکترهای مختلف است در مد RTU بصورت زیر است. همانطور که مشاهده میشود ماکزیمم طول فریم در این مد برابر ۲۵۶ بایت میباشد و ماکزیمم مقدار دیتا ۲۵۳ بایت است.

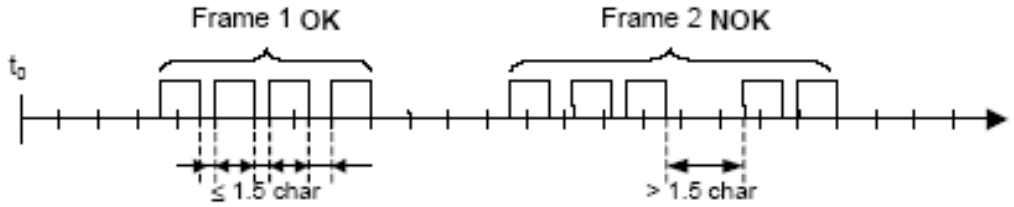
Slave Address	Function Code	Data	CRC
1 byte	1 byte	0 up to 252 byte(s)	2 bytes CRC Low CRC Hi

بین فریم ها یک فاصله زمانی وجود دارد که حداقل به اندازه ۳.۵ کاراکتر است و به آن فاصله خاموشی نیز میگویند.(Silent Interval)





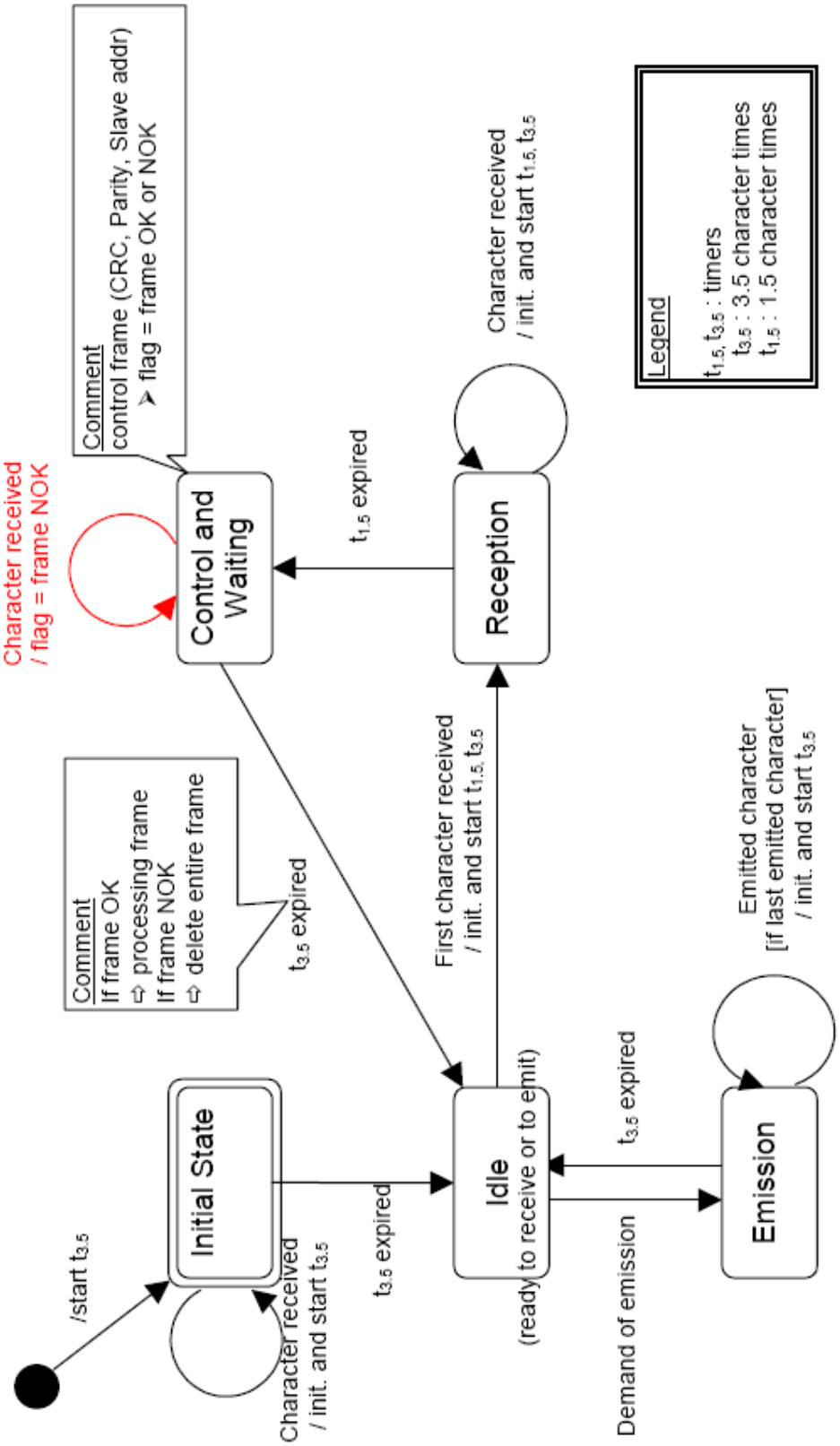
اگر بین دو کاراکتر متوالی یک فریم تاخیری بیش از ۱.۵ کاراکتر پیش بیاید نشان دهنده اشکال است.



این فواصل زمانی که به تایمرهای t۱.۵ و t۳.۵ نیز معروف است برای کنترل نیاز به وقفه های زیاد دارد که در سرعتهای بالا بار زیادی را به CPU تحمیل میکند و بهتر است در سرعتهای بالا زمان این تایمرها فیکس شوند.پیشنهاد میشود برای t۱.۵ زمان ۷۵۰ میکروثانیه و برای t۳.۵ زمان ۱۷۵۰ میلی ثانیه بکار برده شود.

دیاگرام وضعیت مد RTU در شکل بعد آمده است همانطور که مشاهده میشود :

- گذر از وضعیت initial به Idle نیاز به گذشت زمان t۳.۵ دارد تا تاخیر لازم بوجود آید.
- وضعیت Idle وقتی که ارسال یا دریافتی انجام نمیشود بعنوان وضعیت نرمال تلقی میشود.
- وقتی سیستم در وضعیت Idle است هر کاراکتری که پس از آن ارسال شودبعنوان شروع فریم محسوب شده و سیستم را به حالت Active می برد.سپس اگر با گذشت زمان t۳.۵ کاراکتری ارسال نشد بعنوان پایان فریم تلقی می گردد.
- پس از آشکار شدن پایان فریم فیلد CRCچک میشود و بدنبال آن فیلد آدرس آنالیز میشود تا مشخص شود که فریم مربوط به همین وسیله است یا خیر. اگر تطابق نداشت فریم حذف میشود. برای کاهش زمان پردازش فیلد آدرس میتواند به محض دریافت فریم و بدون انتظار برای پایان آن آنالیز شودبدیهی است CRC درصورتی محاسبه و چک میشود که فریم مربوط به همین وسیله باشد.



RTU transmission mode state diagram

چک CRC یا Cyclic Redundancy Check

فیلد CRC محتویات داخل پیام را چک میکند و مستقل از چک Parity مربوط به کاراکترها عمل نماید. CRC از ۱۶ بیت یعنی دو بایت استفاده می کند که این دوبایت به انتهای پیام افزوده میشوند. مقدار CRC به روشی که بیان خواهد شد در موقع ارسال توسط فرستنده محاسبه و به پیام اضافه میشود گیرنده نیز پس از دریافت پیام مقدار CRC را به همان روش از نو محاسبه میکند. اگر مقدار محاسبه شده با آنچه در فریم دیتا موجود است تناقض داشت سیگنال خطا آشکار میگردد. CRC بر اساس ۸ بیت دیتای موجود در بسته دیتا ساخته میشود و به بیتهای Start و Stop و Parity کاری ندارد. بطور خلاصه مراحل ایجاد CRC بصورت زیر است:

۱- بار کردن مقدار FFFF هگز به رجیستر CRC یعنی همه بیتها یک شوند.

۲- XOR کردن ۸ بیت مربوط به بایت پیغام با ۸ بیت مربوط به بایت با ارزش کمتر از ۱۶ بیت رجیستر CRC یعنی ۸ بیت سمت راست و قرار دادن نتیجه در CRC

۳- شیفت داد ن رجیستر CRC یک بیت به راست و وارد کردن صفر از سمت دیگر یعنی سمت MSB یا Most Significant Bit

۴- چک کردن آنچه از LSB یا Least Significant Bit بیرون می افتد .

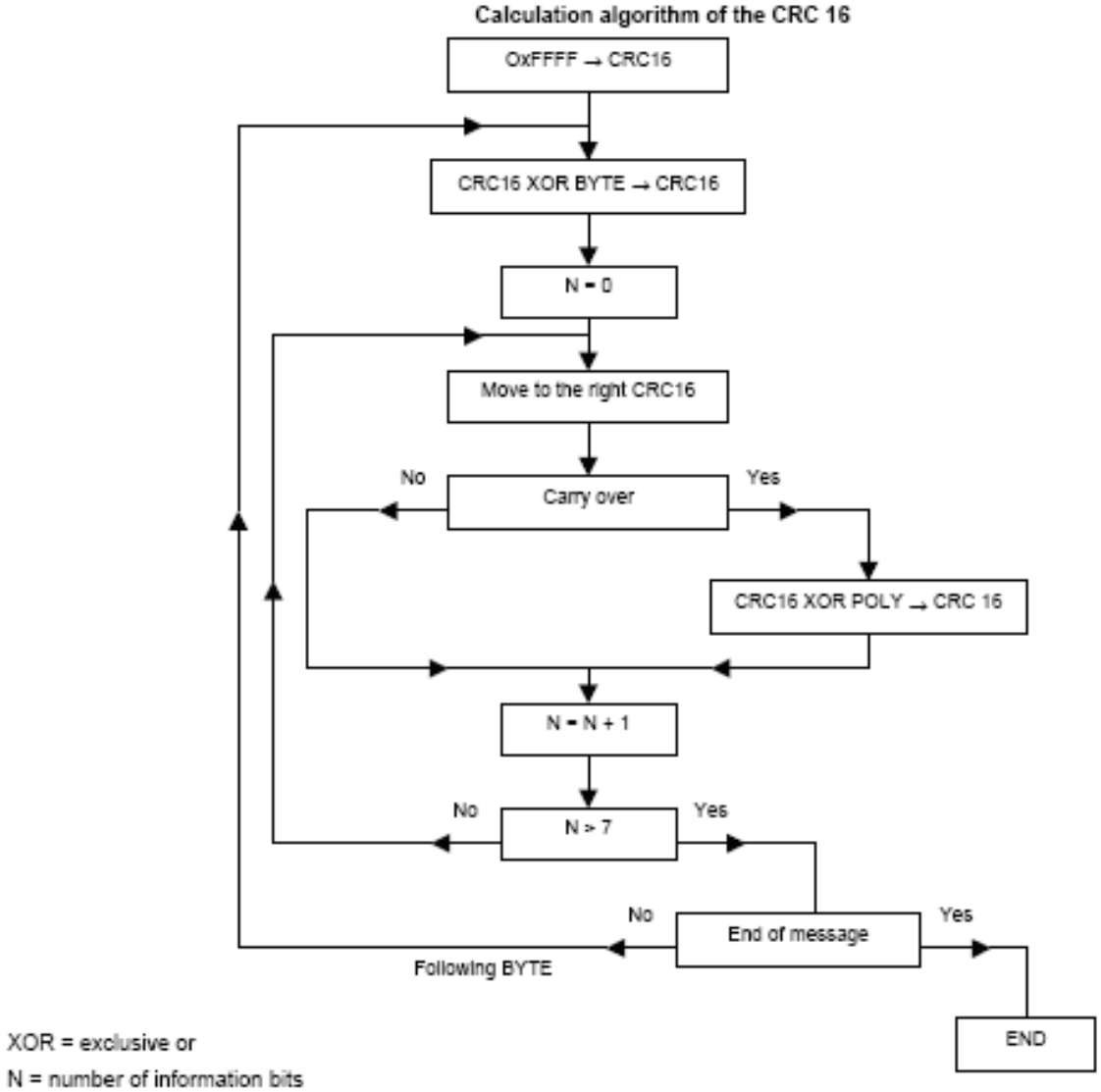
۵- اگر LSB صفر بودتکرار قدم سوم. و اگر LSB یک بود XOR کردن CRC با مقدار هگز ۰XA۰۰۱ یعنی با ۱۰۱۰ ۰۰۰۰ ۰۰۰۰ ۰۰۰۱

۶- تکرار قدم های ۲ تا ۵ برای ۸ بیت بعدی از بایت های پیام تا اینکه اینکار روی تمام بایت ها انجام شود.

نهایتاً آنچه در رجیستر CRC باقی می ماندهمان مقدار محاسبه شده CRC است. پس از اتمام این کار بایتهای بالا و پایین CRC جابجا شده و به فریم پیام اضافه میگردند. در مثال شکل زیر مقدار CRC برابر با ۱۲۴۱ هگز بوده است که با جابجا شدن دوبخش آن ابتدا مقدار ۴۱ و سپس مقدار ۱۲ به پیام اضافه گردیده است.

Addr	Func	Data Count	Data	Data	Data	Data	CRC Lo	CRC Hi
							0x41	0x12

بطور خلاصه الگوریتم محاسبه CRC در صفحه بعد آورده شده و برنامه ای نیز بعنوان نمونه برای محاسبه آن ذکر گردیده است. باز برای روشن شدن دقیقتر موضوع مثالی در ادامه آورده شده است.



XOR = exclusive or

N = number of information bits

POLY = calculation polynomial of the CRC 16 = 1010 0000 0000 0001

BEGIN
ErrorWord = Hex (FFFF)
FOR Each byte in message
ErrorWord = ErrorWord XOR byte in message
FOR Each bit in byte
LSB = ErrorWord AND Hex (۰۰۰۱)

IF LSB = ۱ THEN ErrorWord = ErrorWord – ۱

ErrorWord = ErrorWord / ۲

IF LSB = ۱ THEN ErrorWord = ErrorWord XOR Hex (A۰۰۱)

NEXT bit in byte
NEXT Byte in message

END

CRC register initialization		1111	1111	1111	1111
XOR 1st character		0000	0000	0000	0000
		1111	1111	1111	1101
	Move 1	0111	1111	1111	1110|1
		1010	0000	0000	0001
Flag to 1, XOR polynomial		1101	1111	1111	1111
	Move 2	0110	1111	1111	1111|1
Flag to 1, XOR polynomial		1010	0000	0000	0001
		1100	1111	1111	1110
	Move 3	0110	0111	1111	1110|0
	Move 4	0011	0011	1111	1111|1
		1010	0000	0000	0001
		1001	0011	1111	1110
	Move 5	0100	1001	1111	1111|0
	Move 6	0010	0100	1111	1111|1
		1010	0000	0000	0001
		1000	0100	1111	1110
	Move 7	0100	0010	0111	1111|0
	Move 8	0010	0001	0011	1111|0
		1010	0000	0000	0001
		1000	0001	0011	1110
		0000	0000	0000	0111
XOR 2nd character		1000	0001	0011	1001
	Move 1	0100	0000	1001	1100|1
		1010	0000	0000	0001
		1110	0000	1001	1101
	Move 2	0111	0000	0100	1110|1
		1010	0000	0000	0001
		1101	0000	0100	1111
	Move 3	0110	1000	0010	0111|1
		1010	0000	0000	0001
		1100	1000	0010	0110
	Move 4	0110	0100	0001	0011|0
	Move 5	0011	0010	0000	1001|1
		1010	0000	0000	0001
		1001	0010	0000	1000
	Move 6	0100	1001	0000	0100|0
	Move 7	0010	0100	1000	0010|0
	Move 8	0001	0010	0100	0001|0
			Most significant		least significant

The CRC 16 of the frame is then: 4112

مد انتقال ASCII یا Ameriican Standard Code for Information Interchange

در این مد هر ۸ بیت از بایت پیام بصورت ۲ کاراکتر ASCII ارسال میشود از اینرو بازدهی آن نسبت به RTU کمتر است بعنوان مثال در این مد بایت ۰X5B بصورت دو بایت یعنی "۵"=۰X۳۵ و "B"=۰X۴۲ در می آید. این مد در جایی که لینک فیزیکی یا قابلیت های وسیله اجازه استفاده از مد RTU را نمی دهد (بویژه از نظر مدیریت تایمرها) استفاده میگردد. در مد ASCII فرمت ۱۰ بیت هر بسته دیتا بصورت زیر است:

- ۱ بیت برای شروع
- ۷ بیت برای دیتا
- ۱ بیت برای Parity
- ۱ بیت برای Stop

Start	۱	۲	۳	۴	۵	۶	۷	Par	Stop

نوع Parity شبیه آنچه برای RTU ذکر شد زوج بوده ولی انواع دیگر نیز قابل استفاده است.

نحوه ارسال نیز شبیه RTU است یعنی از بیت LSB شروع و تا بیت MSB ادامه می یابد.

فریم پیام در مد ASCII

هر فریم ASCII از بخشهایی مانند شکل زیر تشکیل شده است:

Start	Address	Function	Data	LRC	End
1 char :	2 chars	2 chars	0 up to 2x252 char(s)	2 chars	2 chars CR,LF

کاراکتر شروع یک کاراکتر که Colon (یعنی :) است و هگز آن ۳A است می باشد. تجهیزات متصل به باس مرتباً باس را برای یافتن این کاراکتر مانیتور می کنند. وقتی این کاراکتر دریافت شد وسیله کاراکتر بعدی را می گیرد که آدرس را مشخص میکند و اگر آدرس به او مربوط بود سایر کاراکترها را دریافت میکند تا بسته به پایان برسد . کاراکتر پایانی Carriage Retun-Line Feed یا CRLF است که کد اسکی آن ۰D و کد هگز آن ۰A میباشد.

بین رشته کاراکترهای داخل پیام ممکن است فاصله زمانی تا یک ثانیه وجود داشته باشد. کاربر با پیکربندی Time out میتواند مشخص کند که اگر زمان بیش از یک ثانیه باشد پیام خطا اعلام شود.برخی از شبکه های وسیع ممکن است به time out بین ۴ تا ۵ ثانیه نیاز داشته باشد.

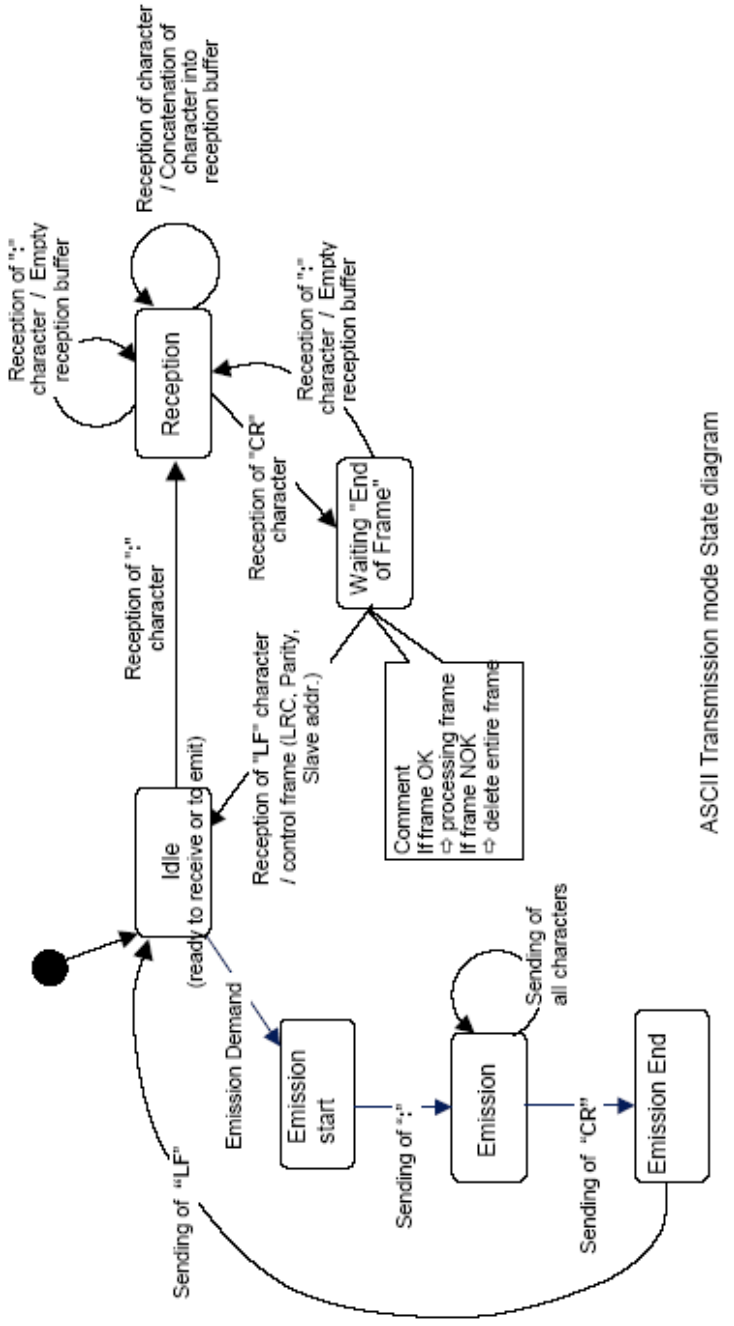
هر بایت دیتا نیاز به ۲ کاراکتر برای کد کردن دیتا داردبنابراین برای اطمینان از سازگاری بین مدهای ASCII و RTU ماکزیمم سایز اختصاص یافته ۲x۲۵۲ کاراکتر است یعنی دو برابرسایز دیتای RTU . بدین ترتیب سایز

کل فریم مطابق شکل بالا ۵۱۳ کاراکتر خواهد بود.کنترل خطا در این مد توسط LRC انجام میشود که تشریح خواهد شد.

دیاگرام وضعیت مد انتقال ASCII

این دیاگرام که هم وضعیت Master و هم وضعیت Slave را نشان میدهد در شکل زیر آمده است.

وضعیت Idle وضعیتی است که در آن هیچ ارسال یا دریافتی روی باس انجام نمیشود. دریافت هر کاراکتر : به معنی شروع پیام جدید است. اگر پیامی هنوز تمام نشده مجدداً کاراکتر : برسد در اینصورت پیام قبلی منتفی تلقی میگردد. با دریافت شدن کاراکتر پایانی یعنی CRLF ، محاسبه و چک LRC شروع میشود.



محاسبه و چک LRC یا Longitudinal Redundancy Check

چک LRC فقط روی خود دیتا انجام میشود پس بیت های شروع و پایان و Parity در محاسبه منظور نمیگردند. LRC یک بایت است (۸ بیت) که بصورت ۲ کاراکتر اسکی در می آید این مقدار در سمت فرستنده به روشی که گفته خواهد شد محاسبه و به پیام اضافه می گردد. گیرنده پس از دریافت پیام نیز شخصاً مبادرت به محاسبه LRC به همان روش میکند و در صورتی که مقدار محاسبه شده با LRC موجود در بسته

یکی بود آنرا بعنوان یک پیام صحیح می پذیرد.LRC با جمع کردن تمام بایتهای داخل پیام بدون بیت های ذکر شده وسپس متمم دو کردن نتیجه محاسبه میشود طبق قدم های زیر:

۱- جمع کردن بایتهای پیام و ریختن آنها در فیلد ۸ بیتی بدیهی است اگر Carry تولید شود از بین میرود.

۲- کم کردن مقدار منتجه از مقدار هگز FF یعنی از ۱۱۱۱ ۱۱۱۱ برای تولید متمم یک

۳- اضافه کردن یک به مقدار حاصل از قدم دوم تا متمم دو بدست آید.

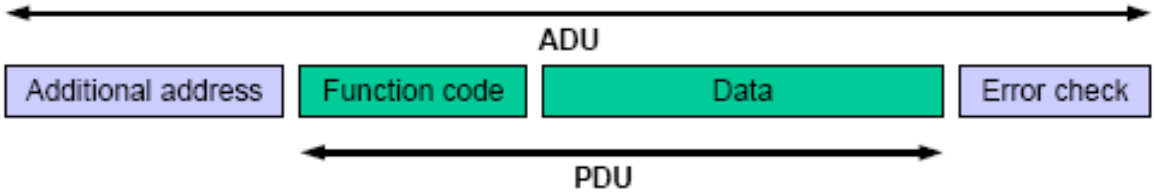
مقدار منتجه بصورت دو کاراکتر به پیام اضافه میشود. در اینحالت برخلاف CRC ابتدا کاراکتر با ارزش بالاتر و سپس کاراکتر با ارزش پایین تر بعد از دیتا قرار میگیرد مثلاً اگر مقدار منتجه ۶۱ هگز یعنی ۰۱۱۰ ۰۰۰۱ باشد شکل زیر را خواهیم داشت:

Colon	Addr	Func	Data Count	Data	Data	Data	Data	LRC Hi	LRC Lo	CR	LF
								"6" 0x36	"1" 0x31		

فانکشن کدهای Modbus

در ارتباط Modbus نوع عملی که باید انجام شود توسط فانکشن کدها مشخص میگردد. بعنوان مثال وقتی Master که بعضاً به آن Client هم گفته میشود بخواهد از روی Slave که ممکن است Server هم نامیده شود I/O خاصی را بخواند این عمل را توسط کد خاصی که در فیلد فانکشن کد از فریم دیتا قرار می دهد درخواست میکند. این کد برای Slave نیز شناخته شده است از اینرو اطلاعات مورد نظر را با همان فانکشن کد به Master بر میگرداند.

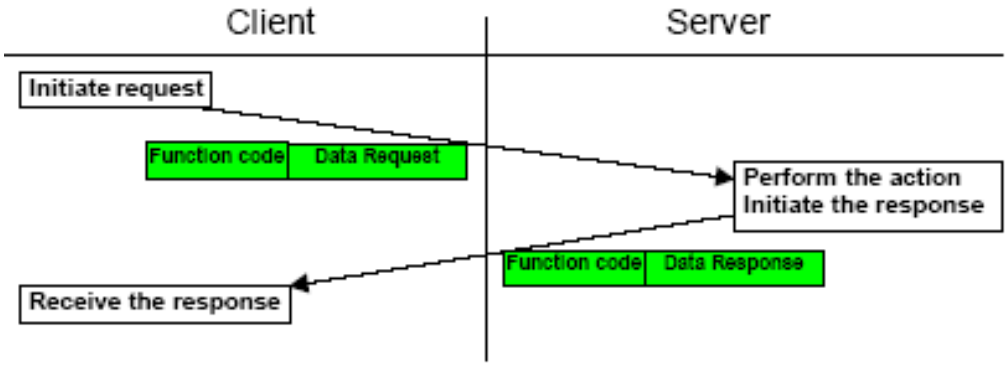
قبل از اینکه به تشریح فانکشن کدها بپردازیم لازم است نحوه تبادل یا Transaction دیتا روی شبکه Modbus تا حدی روشن شودتا جایگاه فانکشن کدها بهتر مشخص گردد. همانطور که قبلاً نیز اشاره شد در فریم دیتا علاوه بر فیلد دیتا فیلد های دیگری نیز وجود دارند. اگر فیلدهای شروع و پایان وفیلد آدرس را در نظر نگیریم با بسته PDU مخفف Protocol Data Unit و بسته ADU مخففApplication Data Unit مانند شکل زیر سرو کار داریم.

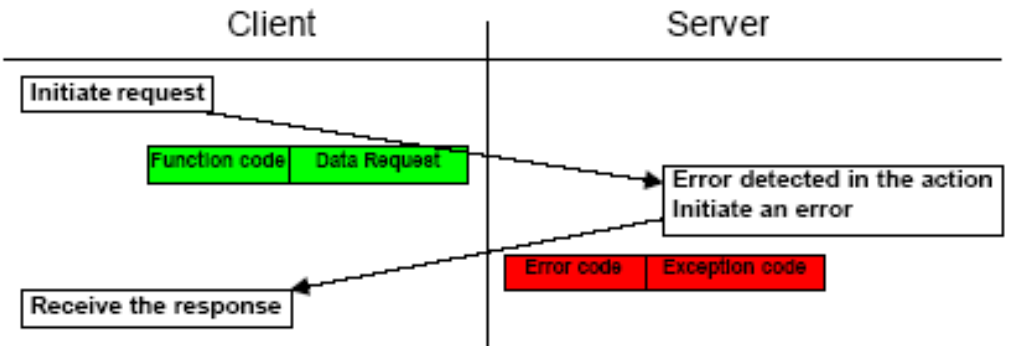


فیلد Function Code یک بایت است که می تواند بین ۱ تا ۲۵۵ دسیمال باشد. کدهای ۱ تا ۱۲۷ برای کار نرمال و کدهای ۱۲۸ تا ۲۵۵ برای پاسخ های Exception که برای شرایط خطا طراحی شده بکار میرود. فیلد Data در برخی درخواست های خاص ممکن است خالی باشد (طول صفر) این در مواردی است که ارسال Action به تنهایی برای Slave کفایت میکند و اطلاعات اضافی مورد نیاز نیست.

وقتی خطایی وجود نداشته باشد Slave درخواست Master را انجام داده و در پاسخ خود به همان کد فانکشن اشاره میکند اصطلاحاً گفته میشود که کدفانکشن **اکو** شده است. ولی وقتی اشکالی وجود داشته باشد و Slave نتواند عمل مورد درخواست Master را انجام دهد در این حالت پاسخی که در آن بجای کدفانکشن کد Exception آمده برگشت داده میشود تا Master از بروز خطا مطلع شود.

شکل های بعد این دو حالت را نمایش میدهد.





با توجه به توضیحات فوق و توجه به نوع فانکشن ها متوجه می شویم که در پروتکل Modbus سه نوع PDU زیر وجود دارد:

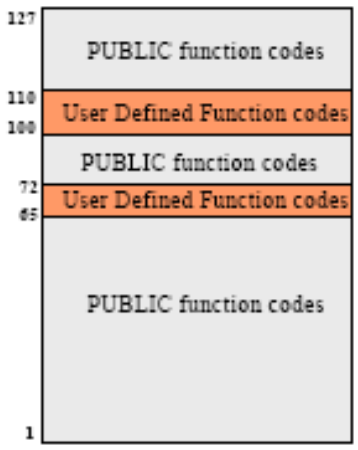
- Request PDU
- Response PDU
- Exception Response PDU

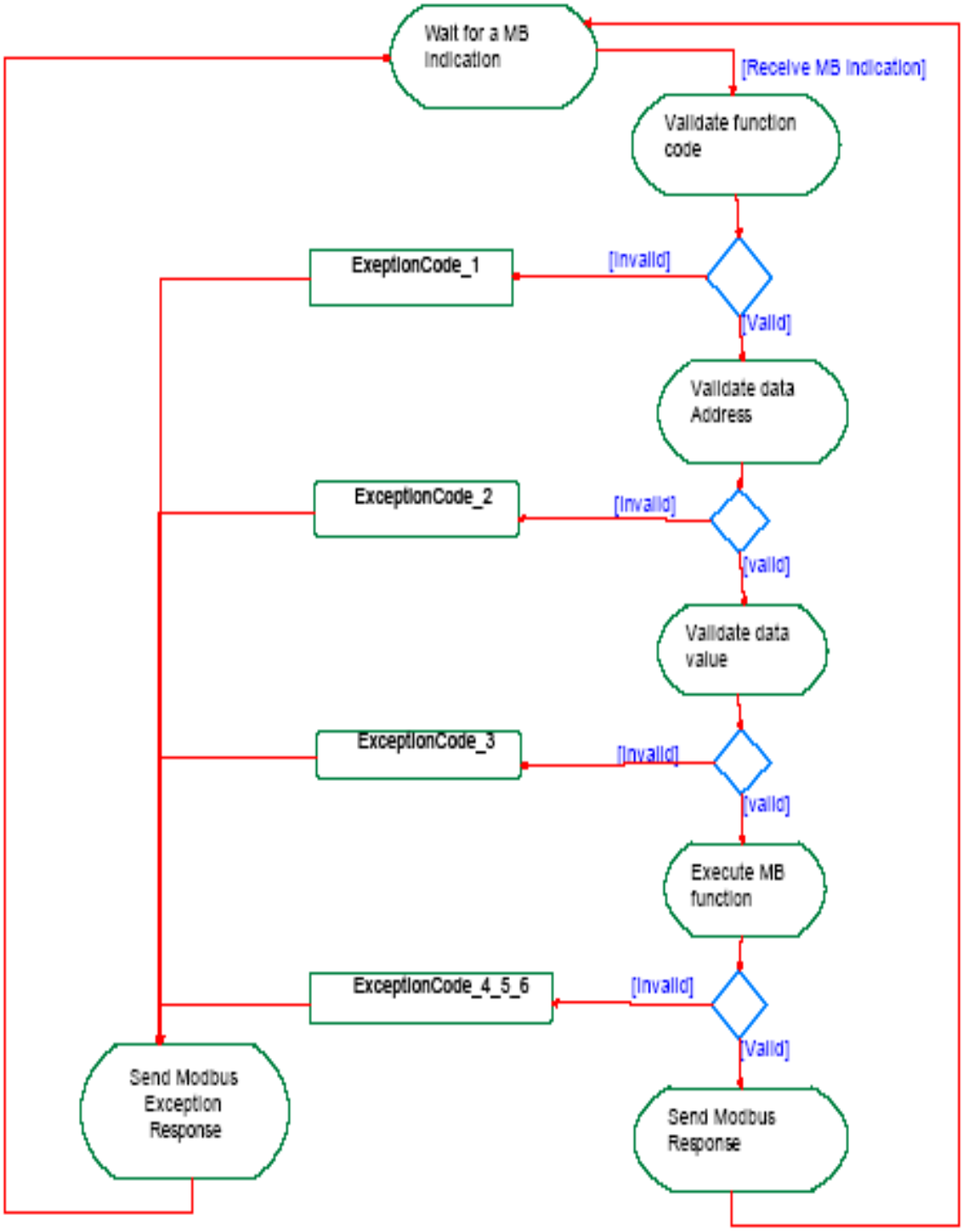
بر این اساس دیاگرام تبادل دیتا روی Modbus بصورت نشان داده در شکل صفحه بعد خواهد بود.

فانکشن کد ها را میتوان به ۲ دسته کلی تقسیم کرد:

۱- فانکشن های عمومی (Public) این فانکشن ها بصورت استاندارد از قبل تعریف شده هستند و برای مقاصد مشخص بکار میروند.

۲- فانکشن های خاص (User Defined) این فانکشن ها می توانند توسط کاربر تعریف شوند و نیازی به تایید موسسه Modbus.org ندارند ولی باید توجه داشت که کد های رزرو شده را برای این فانکشنها نمیتوان استفاده کرد. کد فانکشنهای کاربر میتواند در محدوده ۶۵ تا ۷۲ یا ۱۰۰ تا ۱۱۰ باشد.




Wait for a MB Indication
[Receive MB Indication]
Validate function code
ExeptionCode_1
[Invalid]
[Valid]
Validate data Address
ExceptionCode_2
[Invalid]
[valid]
Validate data value
ExceptionCode_3
[Invalid]
[valid]
Execute MB function
ExceptionCode_4_5_6
[Invalid]
[Valid]
Send Modbus Exception Response
Send Modbus Response

فانکشنهای عمومی

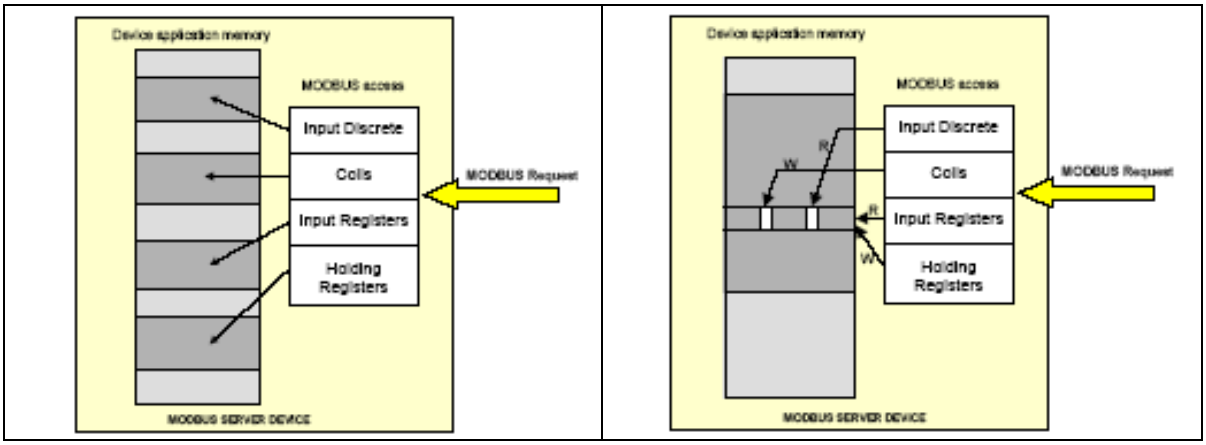
شکل زیر فانکشنهای عمومی قابل استفاده در Modbus را نشان می دهد که برخی از آنها در ادامه تشریح خواهند شد.

			Function Codes			
			code	Sub code	(hex)	Page
Data Access	Bit access	Physical Discrete Inputs	Read Input Discrete	02		02	11
		Internal Bits Or Physical coils	Read Coils	01		01	10
			Write Single Coil	05		05	16
			Write Multiple Coils	15		0F	37
	16 bits access	Physical Input Registers	Read Input Register	04		04	14
		Internal Registers Or Physical Output Registers	Read Multiple Registers	03		03	13
			Write Single Register	06		06	17
			Write Multiple Registers	16		10	39
			Read/Write Multiple Registers	23		17	47
			Mask Write Register	22		16	46
	File record access		Read File record	20	6	14	42
			Write File record	21	6	15	44
Encapsulated Interface			Read Device Identification	43	14	2B	

هر وسیله که روی شبکه Modbus قرار میگیرد دارای یک جدول آدرس بصورت زیر است:

Primary tables	Object type	Type of access
Discretes Input	Single bit	Read-Only
Coils	Single bit	Read-Write
Input Registers	۱۶-bit word	Read-Only
Holding Registers	۱۶-bit word	Read-Write

در حافظه وسیله ممکن است موارد فوق در یک یا چند بلاک بسته به نوع وسیله موجود باشند شکل های زیر:



فانکشن کد ۰۱ یا Read Coils

این فانکشن برای خواندن وضعیت ۱ تا ۲۰۰۰ خروجی از روی وسیله متصل به باس شبکه بکار میرود. در بسته PDU علاوه بر کد ۰۱ آدرس اولین خروجی و تعداد آنها داده می شود. آدرس خروجی از صفر شروع می شود یعنی مثلاً خروجی های ۱ تا ۱۶ بصورت ۰ تا ۱۵ آدرس می گیرند.بنابر این PDU مربوط به درخواست بصورت شکل زیر خواهد بود:

Request

Function code	۱ Byte	۰x۰۱
Starting Address	۲ Bytes	۰x۰۰۰۰ to ۰xFFFF
Quantity of coils	۲ Bytes	۱ to ۲۰۰۰ (۰x۷D۰)

در پاسخ وضعیت خروجی که ۱ برای ON و ۰ برای Off است توسط بیتهای دیتا مشخص میگردد. بیت LSB وضعیت خروجی با آدرس پایین تر را مشخص میکند و همین طور سایر بیتها بترتیب وضعیت سایر خروجی ها را مشخص می نمایند. اگر تعداد خروجی ها مضربی از ۸ نباشد در اینصورت در بایت آخر سایر بیتهای باقی مانده با صفر پر میشوند ولی از آنجا که تعداد خروجی ها نیز برگردانده می شود Master متوجه میشود که تا کجا مربوط به خروجی هاست.

Response

Function code	۱ Byte	۰x۰۱
Byte count	۱ Byte	N*
Coil Status	n Byte	n = N or N+۱

*N= Quantity of Outputs / ۸, if the remainder is different of ۰ □N = N+۱

اگر Error پیش بیاید کد ها بصورت زیر خواهند بود:

Function code	۱ Byte	Function code + ۰x۸۰
Exception code	۱ Byte	۰۱ or ۰۲ or ۰۳ or ۰۴

مثال ۱ : در جدول صفحه بعد مثالی برای خواندن خروجی های ۲۰ تا ۳۸ یعنی جمعاً ۱۹ خروجی آمده است. با توجه به پاسخ مشاهده می کنیم که نتیجه در سه بایت ریخته شده است:

بایت اول: نشان دهنده آدرس های ۲۰-۲۷ از چپ به راست است که با توجه به وضعیت خروجی ها ۱۱۰۰۱۱۰۱ یعنی معادل CD هگز بوده است در این بایت بیت MSB وضعیت خروجی ۲۷ و بیت LSB وضعیت خروجی ۲۰ را نشان می دهد.

Request		Response	
Field Name	*(Hex)*	*Field Name*	*(Hex)*
Function	۰۱	Function	۰۱
Starting Address Hi	۰۰	Byte Count	۰۳
Starting Address Lo	۱۳	Outputs status ۲۷-۲۰	CD
Quantity of Outputs Hi Quantity of Outputs Lo	۰۰	Outputs status ۳۵-۲۸ Outputs status	۶B
	۱۳	۳۸-۳۶	۰۵

بایت دوم : نشان دهنده آدرس های ۲۸-۳۵ از چپ به راست است . که با توجه به وضعیت خروجی ها معادل هگز آن ۶B است.

بایت سوم : نشان دهنده آدرس های ۳۶-۳۸ که فقط ۳ بیت هستند می باشد سایر بیت ها با صفر پر شده اند و معادل هگز ۸ بیت عدد ۰۵ است یعنی ۰۰۰۰ ۰۱۰۱

فلوچارت صفحه بعد نحوه عملکرد فانکشن کد ۰۱ را برای خواندن خروجی نشان می دهد همانطور که ملاحظه میشود در هر مرحله اگر خطایی پیش بیاید کد Exception مربوطه گزارش میشود.

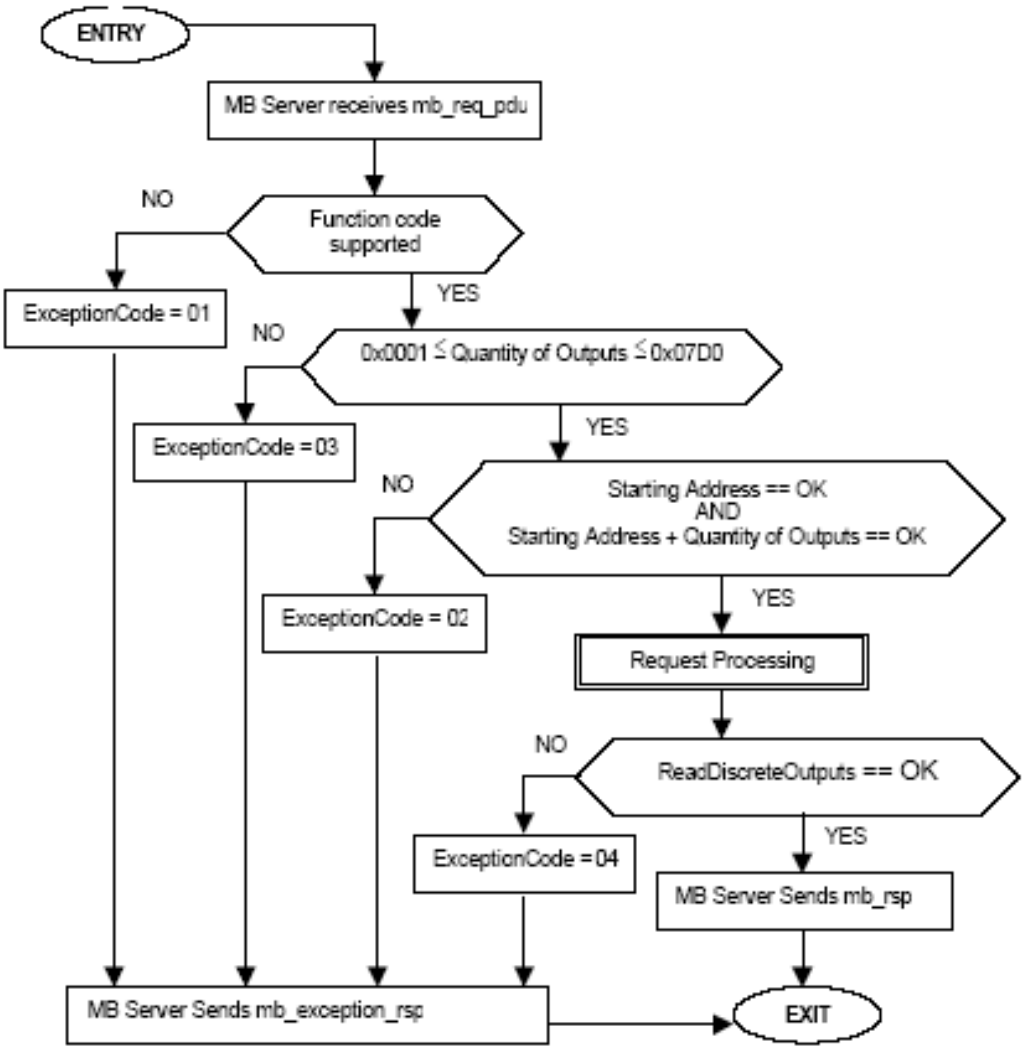
مثال ۲ : در این مثال Master وضعیت دو خروجی یکی خروجی هگز ۰۰۰A که خروجی یازدهم است (چون از صفر شروع شده) و نیز وضعیت خروجی ۰۰۰B که خروجی دوازدهم است را از Slave که آدرس آن روی باس ۰۱ است درخواست میکند . Slave در پاسخ خود اعلام می نماید که این دو خروجی روشن هستند یعنی کد ۰۰۱۱ ۰۰۰۰ که معادل هگز آن ۰۳ است. بدیهی است اگر مثلاً پاسخ کد هگز ۰۲ بود مفهوم آن خاموش بودن اولی و روشن بودن دومی میشد.

Request Message

Address	Function Code	Initial Coil Offset		Number of Points		CRC
		Hi	Lo	Hi	Lo	
01	01	00	0A	00	02	9D C9

Response Frame

Address	Function Code	Byte Count	Coil Data	CRC
01	01	01	03	11 89



فانکشن کد ۰۲ یا Read Input Discrete

این فانکشن برای خواندن تعداد ۱ تا ۲۰۰۰ ورودی دیجیتال از روی وسیله متصل به باس شبکه استفاده میشود. وضعیت آدرس ها و بیت ها شبیه آنچه برای خروجی ها در فانکشن کد ۰۱ ذکر شد می باشد. مثال زیر درخواست و پاسخ را برای خواندن ورودی های ۲۱۸– ۱۹۷ نشان میدهد.

Request		**Response**	
Field Name	*(Hex)*	*Field Name*	*(Hex)*
Function	۰۲	Function	۰۲
Starting Address Hi	۰۰	Byte Count	۰۳
Starting Address Lo	C۴	Inputs Status ۲۰۴-۱۹۷	AC
Quantity of Inputs Hi Quantity of Inputs Lo	۰۰	Inputs Status ۲۱۲-۲۰۵ Inputs Status	DB
	۱۶	۲۱۸-۲۱۳	۳۵

فانکشن کدهای ۰۳ و ۰۴ نیز برای خواندن ورودی ها و خروجی ها از روی رجیستر وسیله بکار میروند که اصول کار آنها با آنجه برای کد فانکشن های ۰۱ و ۰۲ گفته شد یکی است.لذا از تشریح آنها خود داری می کنیم و صرفاً به ذکر مثالی برای آنها بسنده می نماییم.

مثال زیر خواندن آدرس های ۱۰۸ تا ۱۱۰ رجیستر را توسط Master با فانکشن کد ۰۳ نشان می دهد.

Request		Response	
Field Name	*(Hex)*	*Field Name*	*(Hex)*
Function	۰۳	Function	۰۳
Starting Address Hi	۰۰	Byte Count	۰۶
Starting Address Lo	۶B	Register value Hi (۱۰۸)	۰۲
No. of Registers Hi	۰۰	Register value Lo (۱۰۸)	۲B
No. of Registers Lo	۰۳	Register value Hi (۱۰۹)	۰۰
		Register value Lo (۱۰۹)	۰۰
		Register value Hi (۱۱۰) Register	۰۰
		value Lo (۱۱۰)	۶۴

محتویات رجیستر ۱۰۸ با دو بایت که در آن مقدار ۰۲۲B هگز آمده گزارش شده که معادل ۲۵۵ دسیمال است. محتویات رجیستر ۱۰۹معادل ۰۰۰۰ هگز و رجیستر ۱۱۰ معادل ۰۰۶۴ هگز یعنی ۱۰۰ دسیمال گزارش گردیده است.

مثال زیر خواندن Input Register شماره ۹ را توسط فانکشن کد ۰۴ نشان می دهد . محتویات این رجیستر بصورت دو بایت حاوی ۰۰۰A هگز یعنی عدد ۱۰ دسیمال گزارش شده است.

Request		Response	
Field Name	*(Hex)*	*Field Name*	*(Hex)*
Function	۰۴	Function	۰۴
Starting Address Hi	۰۰	Byte Count	۰۲
Starting Address Lo	۰۸	Input reg ۹ Hi	۰۰
Quantity of Input Reg. Hi	۰۰	Input Reg ۹ Lo	۰A
Quantity of Input Reg. Lo	۰۱		

فانکش کد ۰۵ یا نوشتن روی یک خروجی

توسط این فانکشن میتوان یک خروجی را روی وسیله Slave روشن یا خاموش کرد. برای ON کردن مقدار FF۰۰ و برای خاموش کردن مقدار ۰۰۰۰ ارسال می گردد.

Request

Function code	۱ Byte	۰x۰۵
Output Address	۲ Bytes	۰x۰۰۰۰ to ۰xFFFF
Output Value	۲ Bytes	۰x۰۰۰۰ or ۰xFF۰۰

Response

Function code	۱ Byte	۰x۰۵
Output Address	۲ Bytes	۰x۰۰۰۰ to ۰xFFFF
Output Value	۲ Bytes	۰x۰۰۰۰ or ۰xFF۰۰

Error

Error code	۱ Byte	۰x۸۵
Exception code	۱ Byte	۰۱ or ۰۲ or ۰۳ or ۰۴

مثال ۱ : در مثال زیر توسط فانکشن کد ۰۵ خروجی ۱۷۳ روشن می گردد.

Request		Response	
Field Name	*(Hex)*	*Field Name*	*(Hex)*
Function	۰۵	Function	۰۵
Output Address Hi	۰۰	Output Address Hi	۰۰
Output Address Lo	AC	Output Address Lo	AC
Output Value Hi	FF	Output Value Hi	FF
Output Value Lo	۰۰	Output Value Lo	۰۰

مثال ۲: در مثال زیر Master برای خروجی شماره ۱۱ یعنی ۰A هگز (با توجه به اینکه آدرسها از ۰ منظور میشوند) درخواست خاموش شدن میکند. پاسخ ۰۰ که توسط Slave بر می گردد نشان دهنده خاموش شدن وسیله است.

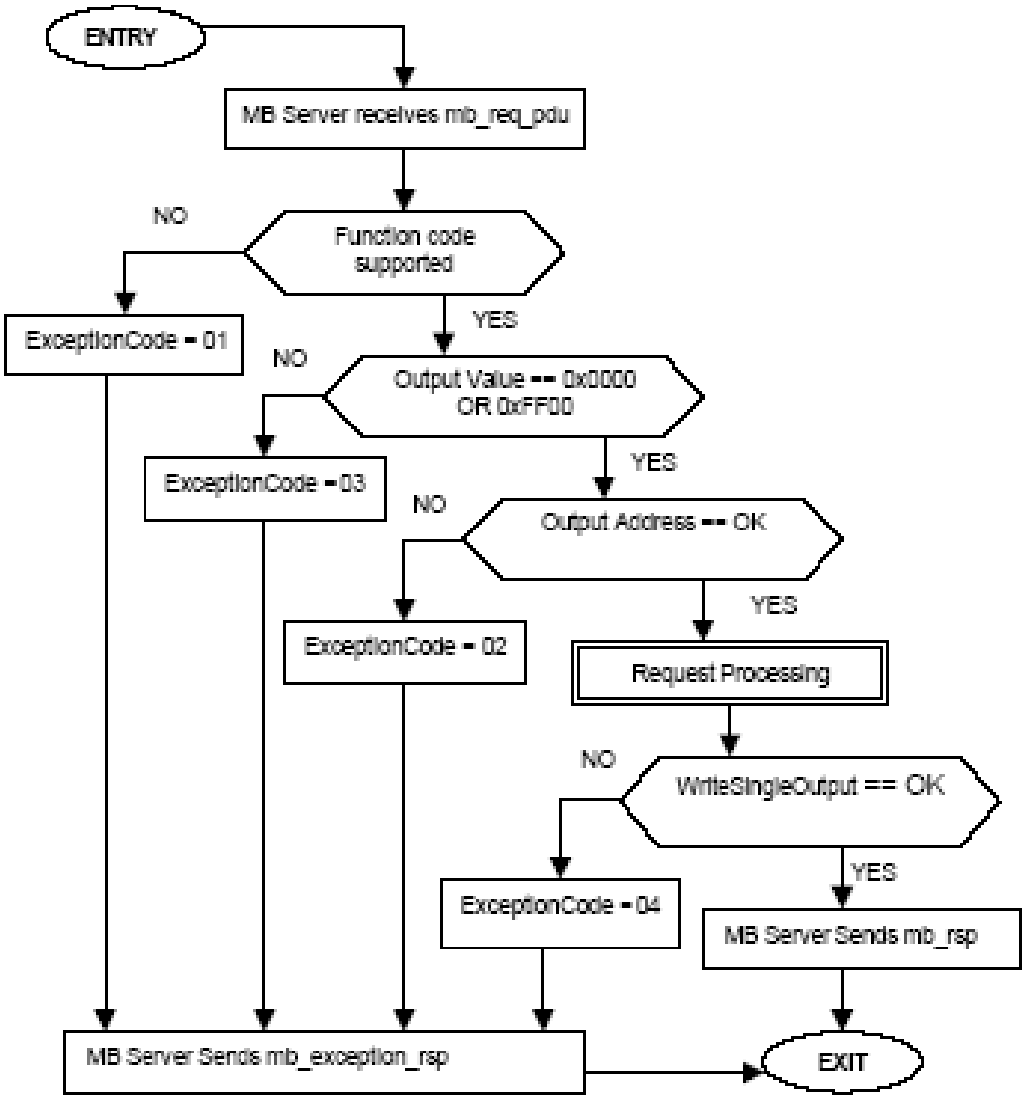
Request Message

Address	Function Code	Coil Offset		New Coil Status		CRC	
		Hi	Lo	Hi	Lo		
01	05	00	0A	00	00	ED	C8

Response Frame

Address	Function Code	Coil Offset		New Coil Status		CRC	
		Hi	Lo	Hi	Lo		
01	05	00	0A	00	00	ED	C8

دیاگرام نوشتن روی خروجی نیز در زیر مثال فوق نشان داده شده است.



فانکشن کد ۰۶ شبیه ۰۵ ولی برای نوشتن روی یک بیت از رجیستر بکار میرود.

فانکشن کد ۱۵ نوشتن روی چندین خروجی

این فانکشن میتواند بطور همزمان وضعیت چندین خروجی را ON یا Off کند . خروجی ها از آدرس ۰ شروع میشوند و با ارسال ۱ منطقی روشن و با ارسال ۰ منطقی خاموش میشوند.

Request PDU

Function code	۱ Byte	۰x۰F
Starting Address	۲ Bytes	۰x۰۰۰۰ to ۰xFFFF
Quantity of Outputs	۲ Bytes	۰x۰۰۰۱ to ۰x۰۷B۰
Byte Count	۱ Byte	N*
Outputs Value	N* x ۱ Byte	

*N= Quantity of Outputs / ۸, if the remainder is different of ۰ □N = N+۱

Response

Function code	۱ Byte	۰x۰F
Starting Address	۲ Bytes	۰x۰۰۰۰ to ۰xFFFF
Quantity of Outputs	۲ Bytes	۰x۰۰۰۱ to ۰x۰۷B۰

Error

Error code	۱ Byte	۰x۸F
Exception code	۱ Byte	۰۱ or ۰۲ or ۰۳ or ۰۴

بعنوان مثال برای ارسال مقدار CD۰۱ هگز که معادل ۱۱۰۰۱۱۰۱۰۰۰۰۰۰۰۱ روی ۱۰ خروجی که از آدرس ۲۰ شروع میشوند روش بصورت زیر است:

	CD هگز								۰۱ هگز							
Bit:	۱	۱	۰	۰	۱	۱	۰	۱	۰	۰	۰	۰	۰	۰	۰	۱
Output:	۲۷	۲۶	۲۵	۲۴	۲۳	۲۲	۲۱	۲۰	_	_	_	_	_	_	۲۹	۲۸

توجه شود که مقدار ۰۱ هگز روی آدر سهای ۲۸ و ۲۹ ارسال می گردد و سایر بیتهای استفاده نشده با صفر پر می شوند.

فانکشن کد ۱۶ نیز شبیه فانکشن کد ۱۵ ولی برای نوشتن روی چندین بیت از رجیستر بکار میرود.

فانکشن کد ۲۰ خواندن رکوردهایی از فایل

در رجیستر هر وسیله آدرسهایی از حافظه که به آن ها رکورد میگوییم وجود دارند . مجموعه ای از رکوردها (تا ۱۰۰۰۰ رکورد) یک فایل را تشکیل میدهند این رکوردها از آدرس ۰ تا ۹۹۹۹ دسیمال هستند (از ۰۰۰۰ تا ۲۷۰F هگز). فانکشن کد ۲۰ میتواند چندین گروه از رکوردهای مختلف را از فایلهای مختلف بخواند . برا ی این کار هر گروه نیاز به رفرنس هایی در ۷ بایت بصورت زیر دارد:

- ۱ بایت : نوع
- ۲ بایت : شماره فایل
- ۲ بایت : آدرس شروع رکوردها
- ۲ بایت : طول رکوردها

این موارد همراه با Request ارسال میشوند و به آنها Sub-Request میگویند.

Request PDU

Function code	۱ Byte	۰x۱۴
Byte Count	۱ Byte	۰x۰۷ to ۰xF۵ bytes
Sub-Req. x, Reference Type	۱ Byte	۰۶
Sub-Req. x, File Number	۲ Bytes	۰x۰۰۰۰ to ۰xFFFF
Sub-Req. x, Record Number	۲ Bytes	۰x۰۰۰۰ to ۰x۲۷۰F
Sub-Req. x, Register Length	۲ Bytes	N
Sub-Req. x+۱, ...		

Response PDU

Function code	۱ Byte	۰x۱۴
Resp. data Length	۱ Byte	۰x۰۷ to ۰xF۵
Sub-Req. x, File Resp. length	۱ Byte	۰x۰۷ to ۰xF۵
Sub-Req. x, Reference Type	۱ Byte	۶
Sub-Req. x, Record Data	Nx ۲ Bytes	

Sub-Req. x+۱, ...		

Error

Error code	۱ Byte	۰x۹۴
Exception code	۱ Byte	۰۱ or ۰۲ or ۰۳ or ۰۴ or ۰۸

در مثال زیر دو گروه رکورد با رفرنس های زیر از روی وسیله Remote درخواست میشوند:

- گروه ۱ : شامل دو رجیستر از فایل شماره ۴ که از آدرس ۱ شروع می شوند.
- گروه ۲ : شامل دو رجیستر از فایل شماره ۲ که از آدرس ۹ شروع می شوند.

Request		**Response**	
Field Name	*(Hex)*	*Field Name*	*(Hex)*
Function	۱۴	Function	۱۴
Byte Count	۰C	Resp. Data length	۰E
Sub-Req. ۱, Ref. Type	۰۶	Sub-Req. ۱, File resp. length	۰۵
Sub-Req. ۱, File Number Hi	۰۰	Sub-Req. ۱, Ref. Type	۰۶
Sub-Req. ۱, File Number Lo	۰۴	Sub-Req. ۱, Record. Data Hi	۰D
Sub-Req. ۱, Record number Hi	۰۰	Sub-Req. ۱, Record. Data Lo	FE
Sub-Req. ۱, Record number Lo	۰۱	Sub-Req. ۱, Record. Data Hi	۰۰
Sub-Req. ۱, Record Length Hi	۰۰	Sub-Req. ۱, Record. Data Lo	۲۰
Sub-Req. ۱, Record Length Lo	۰۲	Sub-Req. ۲, File resp. length	۰۵
Sub-Req. ۲, Ref. Type	۰۶	Sub-Req. ۲, Ref. Type	۰۶
Sub-Req. ۲, File Number Hi	۰۰	Sub-Req. ۲, Record. Data Hi	۳۳
Sub-Req. ۲, File Number Lo	۰۳	Sub-Req. ۲, Record. Data Lo	CD
Sub-Req. ۲, Record number Hi	۰۰	Sub-Req. ۲, Record. Data Hi	۰۰
Sub-Req. ۲, Record number Lo	۰۹	Sub-Req. ۲, Record. Data Lo	۴۰
Sub-Req. ۲, Record Length Hi	۰۰		
Sub-Req. ۲, Record Length Lo	۰۲		

فانکشن کد ۲۱ برای نوشتن در رکورهایی از فایل

این فانکشن کد شبیه ۲۰ ولی بجای خواندن کار نوشتن را انجام میدهد

پاسخ های Exception

تاکنون در خلال بحث چندین بار به پاسخ های Exception اشاره و کد هایی را نیز بعنوان Error در ارتباط با هر فانکشن کد ذکر کرده ایم. برای روشن شدن مفهوم اینگونه پاسخ ها حالات زیر را در نظر بگیرید:

- وقتی وسیله درخواستی را از Master بدون خطای ارتباطی دریافت میکند پاسخ دادن به آن بطور نرمال است.
- وقتی وسیله درخواست Master را بدلیل وجود اشکال ارتباطی دریافت نمیکند پاسخی هم نمی دهد. در این حالت در سمت Master خطا توسط Time Out آشکار میشود.
- وقتی وسیله درخواست را دریافت میکند ولی در آن خطا میبیند مثلاً با توجه به Parity و CRC یا LRC متوجه میشود که برخی از بیتها در هنگام انتقال خراب شده اند در اینحالت نیز پاسخی برگشت داده نمیشود و باز Master با Time out متوجه اشکال میشود.
- وقتی وسیله درخواست Master را بدون خطا های فوق دریافت میکند ولی نمی تواند آنرا انجام دهد بعنوان مثال درخواست برای خواندن از آدرس هایی است که عملاً وجود ندارند در این حالت وسیله پاسخ Exception بر میگرداند. پاسخ Exception به این نحو است که در PDU دریافت شده کد فانکشن به همان شکل بر نمی گردد بلکه وسیله بیت MSB آنرا که صفر است به یک تبدیل میکند بعبارت دیگر آنرا با ۸۰ هگز جمع می کند و سپس آنرا به Master بر میگرداند . Master با دریافت این کد متوجه اشکال میشود. اما اینکه چه نوع خطایی رخ داده از روی فیلد Data متوجه میشود زیرا وسیله کد خطای Exception را نیز در این فیلد قرار میدهد. این کد ها طبق شکل زیر هستند:

Code	Name	Description
01	Illegal function	Requested function is not supported
02	Illegal data address	Requested data address is not supported
03	Illegal data value	Specified data value is not supported
04	Failure in associated device	Slave PLC has failed to respond to a message
05	Acknowledge	Slave PLC is processing the command
06	Busy, rejected message	Slave PLC is busy

در مثال زیر درخواست خواندن وضعیت خروجی های ۵۱۴ تا ۵۲۱ توسط Master ارسال میگردد ولی چون این آدرسها توسط وسیله ساپورت نمیشوند درپاسخ کد ۰۲ Exception که نشان دهنده ناشناخته بودن آدرس است را برمیگرداند.

Request Message

Address	Function Code	Starting Point	Number of Points	CRC
01	01	02 01	00 08	6D B4

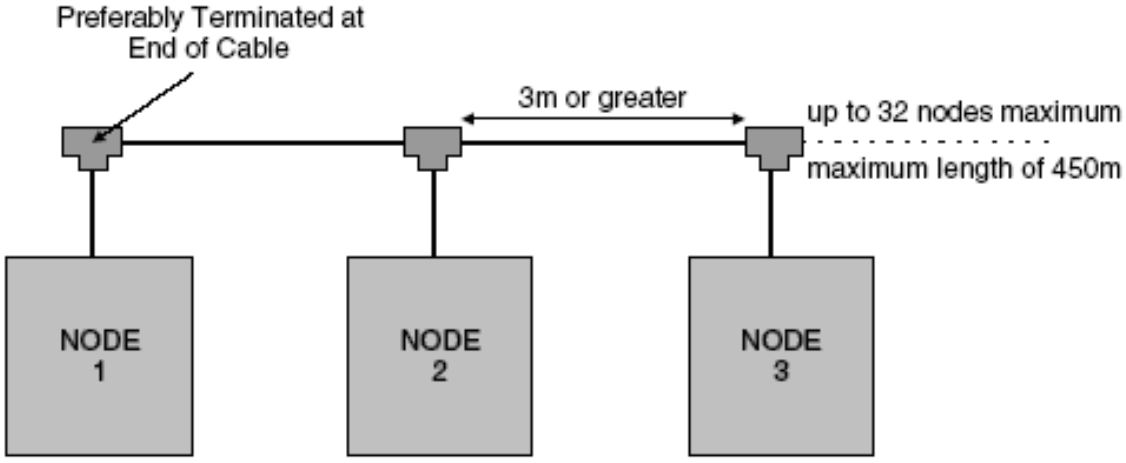
Exception Response Message

Address	Function Code	Exception Code	CRC
01	81	02	C1 91

Modbus Plus ۳-٤

پروتکل Modbus Plus بمنظور فائق آمدن بر محدودیت هایی که در نسخه پایه Modbus وجود داشت از جمله محدودیت Single Master بودن سیستم و مشکلات ساختاری که Modbus در شبکه های وسیع پیدا می کرد عرضه شد. Modbus Plus در واقع جزو پیشگامان استفاده از تکنیک Token Pass روی شبکه بود. ولی برخلاف نسخه پایه بعنوان یک استاندارد باز طرح نشد.

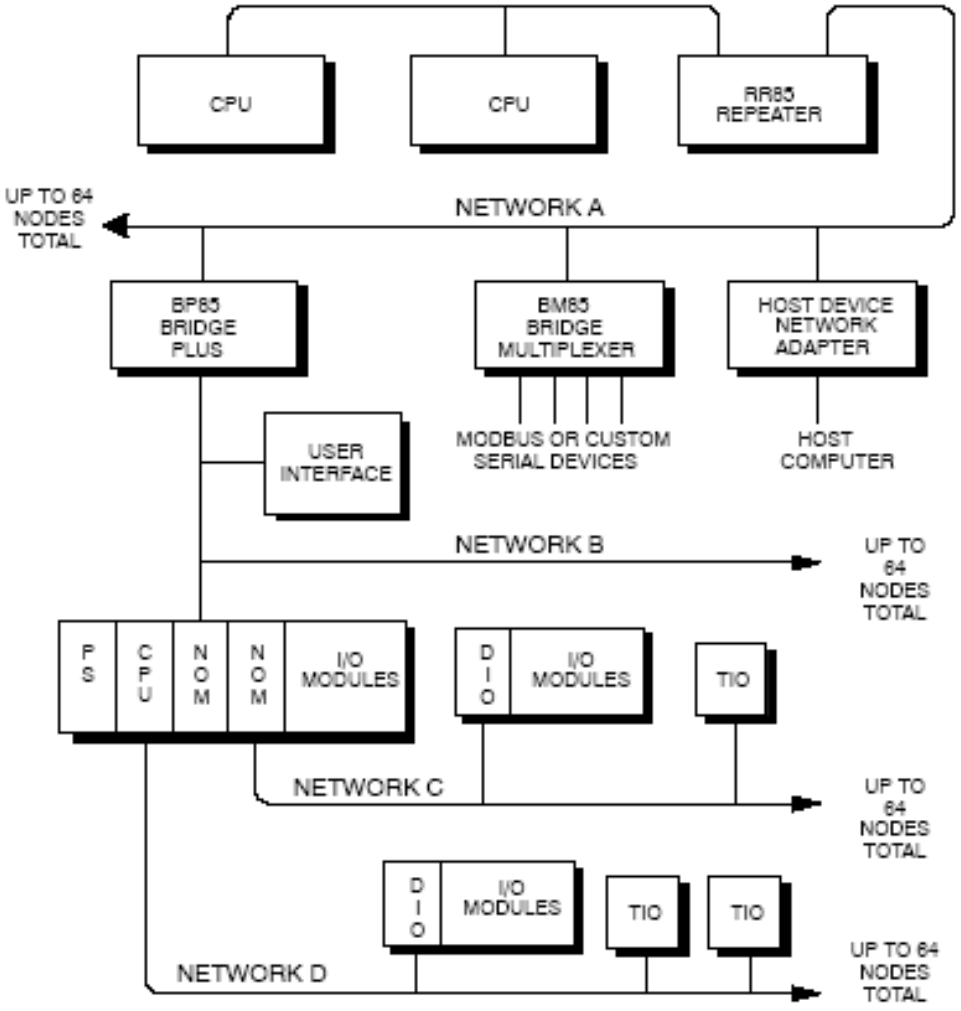
در شبکه Modbus Plus ماکزیمم ۶۴ وسیله (Node) می توان روی باس قرار داد که از این تعداد ۳۲ تا میتوانند بصورت مستقیم به شبکه ای با ماکزیمم طول ۴۵۰ متر متصل شوند. برای اتصال وسایل بیشتر به شبکه یا افزایش طول لازم است از ریپیتر استفاده گردد.که در اینصورت ماکزیمم ۶۴ وسیله روی باس با طول ماکزیمم ۱۸۰۰ متر قابل اتصال است. با بکار بردن فیبر نوری مسافتهای زیادتر نیز پوشش داده میشوند.



میتوان با استفاده از Bridge مخصوص این شبکه که به (Bridge Plus) BP موسوم است چندین شبکه را باهم لینک کرد بعلاوه میتوان با استفاده از Multiplexer های مخصوص که به MP موسوم است و دارای لینک سریال RS۲۳۲/RS۴۸۵ می باشد اتصال با وسایل مجهز به پورت سریال را برقرار کرد.

PLC های Modicon دارای پورت اتصال مستقیم به Modbus Plus هستند و از طریق این پورت میتوان Distributed I/O هایی که شبکه شده اند را به PLC متصل نمود.در عین حال با قرار دادن کارت شبکه که به NOM مخفف Network Option Module است اتصال بین PLC و شبکه مزبور را برقرار نمود.

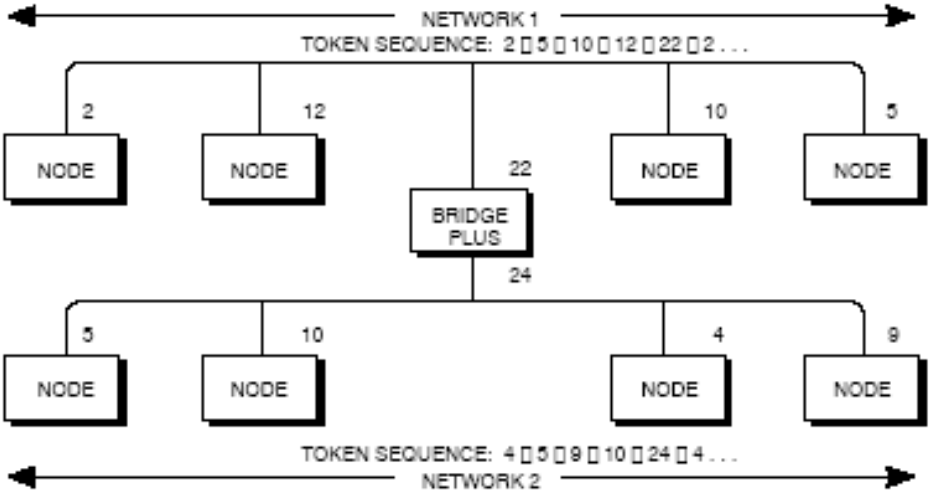
در شکل صفحه بعد نمونه ای از ساختار شبکه Modbus Plus نشان داده شده است. همانطور که ملاحظه میشود شبکه A توسط ریپیتر توسعه پیدا کرده و ارتباط بین شبکه A و شبکه B از طریق Bridge انجام شده است.



٤-٣-١ ارتباط منطقی (Logic) در شبکه Modbus Plus

در شبکه Modbus Plus هر Node دارای یک آدرس بین ۱ تا ۶۴ است که این آدرس الزاماً نباید به ترتیب باشد و ربطی هم به محل نصب فیزیکی وسیله ندارد صرفاً منحصر به فرد بودن آن الزامی است. هر Node زمانی

اجازه دسترسی به باس دارد که فریم Token در اختیارش باشد. Token گروهی از بیت های منطقی است که بین Node ها به ترتیب آدرس آنها می چرخد. هر شبکه دارای Token مخصوص به خود می باشد از اینرو وقتی چندین شبکه Modbus Plus توسط Bridge به یکدیگر متصل میشوند Token در هر قسمت جداگانه می چرخد و Bridge اجازه عبور آن را نخواهد داد. شکل بعد دو شبکه را که با Bridge به هم متصل شده اند نشان میدهد. Token هر شبکه بصورت مجزا به ترتیب شماره Node ها و بصورت صعودی گردش میکند و پس از رسیدن به بالاترین آدرس مجدداً به آدرس پایین تر بر میگردد.



وقتی یک Node بخواهد به تبادل دیتا با سایر Node ها بپردازد ابتدا Token را در اختیار گرفته و آنرا بمدت حداقل ۵۳۰ میکروثانیه نگه میدارد یا اصطلاحاً Hold میکند سپس قادر است پیام خود را ارسال نماید. در هر پیام آدرس گیرنده و فرستنده مشخص است. در شبکه هایی که با Bridge به هم متصل شده اند مسیر (route) نیز در پیام مشخص میگردد. با عبور Token هر Node میتواند در یک دیتابیس سراسری (global) که بصورت Broadcast برای تمام Node ها محسوب میشود پیامی را بنویسد.

دیتای Global بعنوان فیلدی در فریم Token قرار میگیرد و همراه با آن در شبکه آزاد می شود. سایر Node ها با مانیتور کردن Token فیلد Global Data را از آن جدا کرده و پیام را میخوانند. این روش امکان انتقال سریع آلارم ها و Set Point ها و سایر دیتا های مهم را فراهم می سازد . بدیهی است هر شبکه برای خودش دارای یک Global Data است که نمیتواند از Bridge عبور کرده و به سایر شبکه ها منتقل شود.

هر Bridge نیز دارای آدرس Node روی شبکه است ولی از آنجا که واسط بین دو شبکه میباشد میتواند روی هر شبکه یک آدرس Node داشته باشد. در شکل بعد ملاحظه میشود که Bridge بین شبکه A و B از دید شبکه A با آدرس ۲۲ و از دید شبکه B با آدرس ۲۵ شناخته میشود ولی Bridge بین شبکه B و C از دید

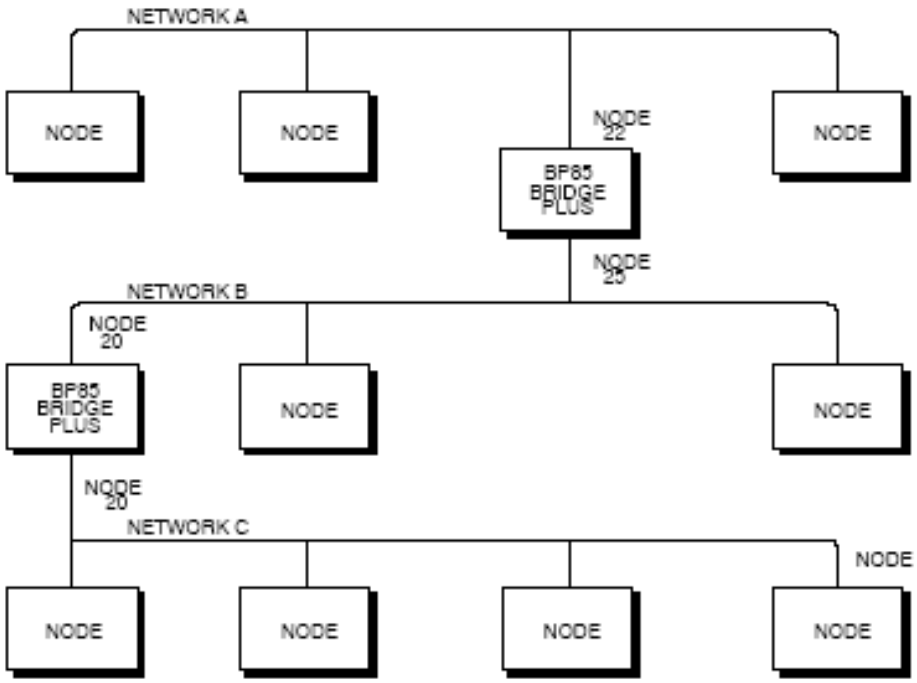
هر دو شبکه با آدرس ۲۰ مشخص میشود . اگر پیامی از Node یک شبکه به Node شبکه دیگری که با Bridge به هم متصل شده اند ارسال شود ابتدا Bridge آنرا دریافت کرده و ذخیره می کند سپس به محض اینکه Token شبکه دوم را گرفت پیام مزبور را همراه با آن به Node مورد نظر ارسال می نماید. هر فریم پیام شامل اطلاعات مسیر (Route) است یعنی آدرس Bridge هایی که لازم است پیام از آنها بگذرد تا به مقصد برسد در فیلدهای مربوطه مشخص است. شکل کلی این فریم در صفحه بعد آمده است. ماکزیمم ۴ آدرس Route میتوان در فریم پیام بکار برد. بعنوان مثال فرض کنید که در شکل صفحه بعد Node شماره ۵ از شبکه A بخواهد پیامی را به Node شماره ۱۲ که در شبکه B قرار دارد بفرستد برای اینکار:

۱- اولین آدرس Route آدرس Bridge بین A و B یعنی ۲۲ میباشد.

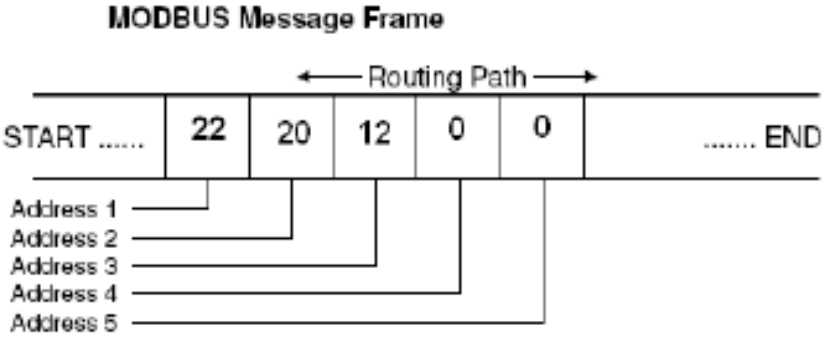
۲- دومین آدرس Route آدرس Bridge بین B و C یعنی ۲۰ میباشد. توجه شود که آدرس Bridge بین A و B از دیدگاه شبکه B آدرس ۲۵ است و این Bridge وقتی Token را بدست بیاورد پیام را به Node شماره ۲۰ روی شبکه B یعنی Bridge دیگر ارسال میدارد.

۳- سومین آدرس Route مربوط به آدرس Node مقصد است .

۴- سایر فیلدهای آدرس Route با صفر پر میشوند یعنی Forwarding بیشتری لازم نیست.



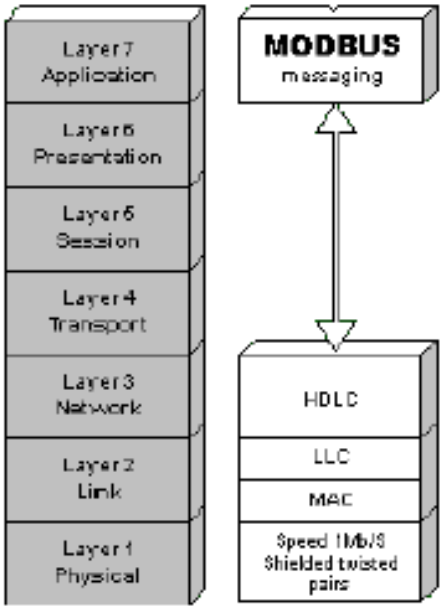
وقتی Bridge اولیه یعنی ۲۲ پیام را میگیرد آنرا بررسی میکند و از آنجا که آدرس Route های بعدی صفر نیست متوجه می شود که باید آنرا به شبکه بعدی بفرستد. این Bridge آدرس خودش را بر می دارد و آدرس Route های موجود در پیام را به چپ شیفت داده و بیتهای سمت راست را با صفر پر میکند بدین ترتیب آدرس ۲۰ در اولین فیلد Route قرار میگیرد . این Bridge با دریافت Token فریم مزبور را به Node شماره ۲۰ ارسال مینماید.عملکرد Bridge شماره ۲۰ برای Routing نیز به همین منوال است.



لایه های پروتکل Modbus Plus

لایه های پروتکل Modbus Plus در مقایسه با مدل OSI در شکل روبرو آمده است همانطور که ملاحظه میشود برای بسته بندی یا باز کردن پیام از سه سطح پروتکلی استفاده میشود که عبارتند از :

- HDLC
- LLC
- MAC



HDLC

HDLC که مخفف High Level Data Link Control است دارای فریمی بصورت شکل زیر است

	PREAMBLE	OPENING FLAG	BDCST ADDRESS	MAC / LLC FIELD	FCS	CLOSING FLAG	
	AA	vE	FF		CRC_۱۶	vE	

طول بایت	۱	۲		۱	۱	۱	۱

هر فریم HDLC متشکل از فیلدهای زیر است:

Preamble : آغاز یا مقدمه فریم را مشخص مکند و مقدار هگز AA می باشد.

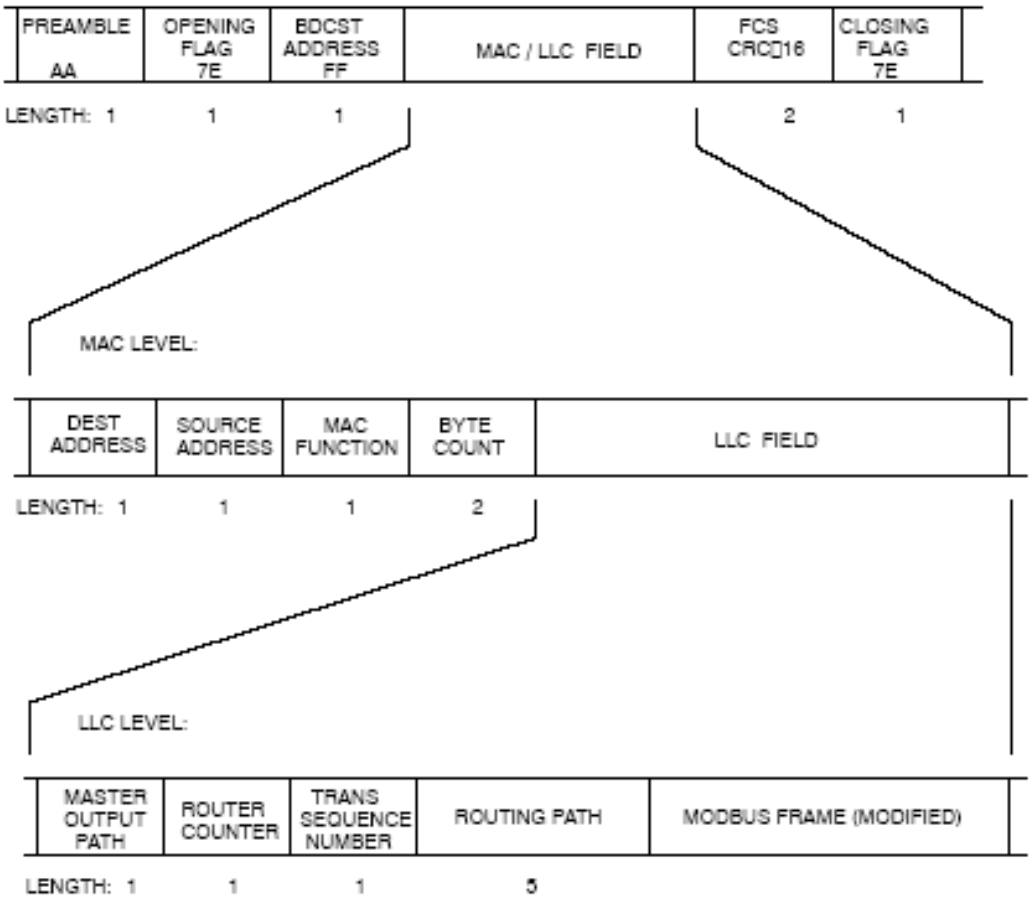
Opening : نشانه ابتدای فریم است ومقدار هگز 7E می باشد.

Broadcast Address :این فیلد مشخص میکند که آیا همه Node ها لازم است پیام را بگیرند یا خیر و مقدار آن FF هگز است.

MAC/LLC : این فیلد حاوی اطلاعات Token و دیتاست و در ادامه تشریح خواهد شد.

FCS : Frame Check Sequence فیلد چک کردن خطاست که بصورت CRC عمل میکند.

Closing : نشانه انتهای فریم است ومقدار هگز 7E می باشد.



MAC Level : Media Access Control Level همانطور که در شکل نشان داده شده است مشتمل بر فیلدهای زیر است:

Destination Address : آدرس مقصد را مشخص میکند که بین ۱ تا ۶۴ است.

Source Address : آدرس مبدا را مشخص میکند و بین ۱ تا ۶۴ است.

MAC Function : کدی است که توسط آن مقصد میفهمد چه کاری باید انجام دهد.

Byte Count : تعداد بایت موجود در بسته را مشخص میکند.

LLC Level

Logical Link Control حاوی پیامی است که باید ارسال شود مانند یک فرمان Modbus بعلاوه این فیلد مسیرهای آدرس Route و سایر اطلاعات مرتبط با پیام را نیز در بر میگیرد. فیلدهای آن عبارتند از :

Master Output Path : مسیر خروجی کنترلر یعنی پورتی که توسط آن ارتباط برقرار میکند را مشخص میکند

Router Counter :تعداد Bridge Plus مسیر را مشخص میکند.

Transaction Sequence : دادو ستد یا تبادل دیتا بین مبدا و مقصد را نشان می دهد.

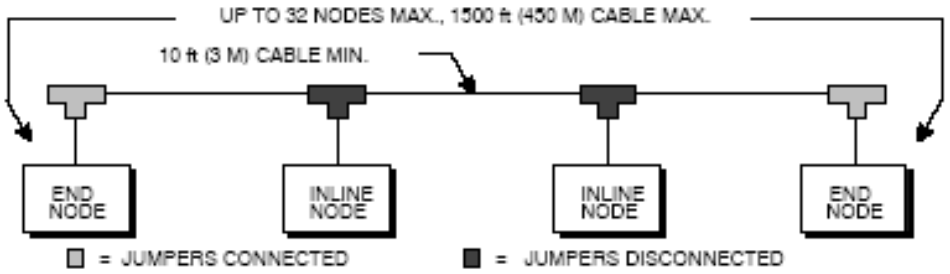
Routing Path : مسیر آدرس را در هنگام استفاده از Bridge مشخص میکند.

Modbus Frame :فرمان های Modbus Plus را دربر دارد که عمدتاً شبیه فرمانهای نسخه پایه Modbus است.

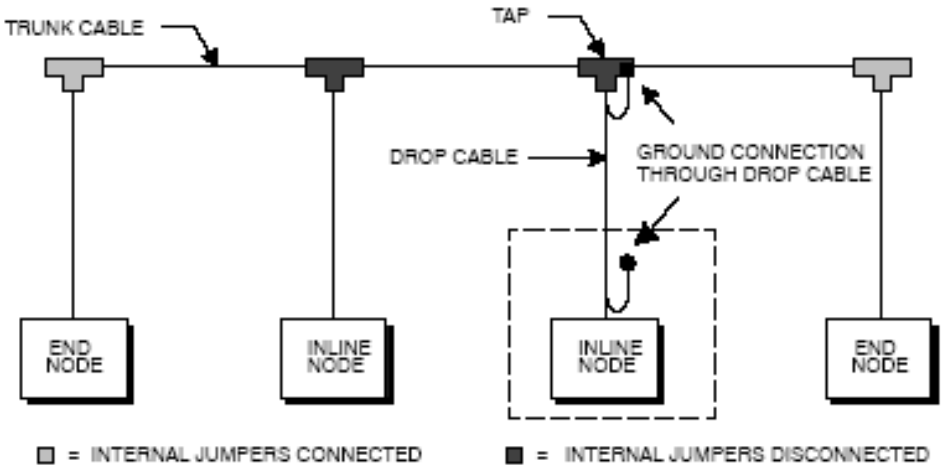
٤-۳-۲ لایه فیزیکی شبکه Modbus Plus

در لایه فیزیکی Node های شبکه توسط یک کابل دو رشته بهم تابیده STP بیکدیگر متصل میشوند در عین حال میتوان از فیبر نوری نیز استفاده کرد. طول کابل مسی در هر Section حداکثر ۴۵۰ متر است اگر نیاز به گسترش شبکه باشد از ریپیتر استفاده میشود بدیهی است طول Section بعدی نیز ۴۵۰ متر خواهد بود . میتوان تعداد ریپیتر ها را آنقدر افزایش داد تا نهایتاً طول کل شبکه به ۱۸۰۰ متر برسد در اینحالت باید توجه داشت که:

- بین هر جفت Node که قرار است با هم ارتباط برقرار کنند نباید بیش از ۳ ریپیتر موجود باشد.
- طول کابل بین دو Node مجاور نباید از ۳ متر کمتر باشد.
- هر Section حداکثر ۳۲ Node میتواند داشته باشد
- تعداد کل Node ها حداکثر ۶۴ عدد است.

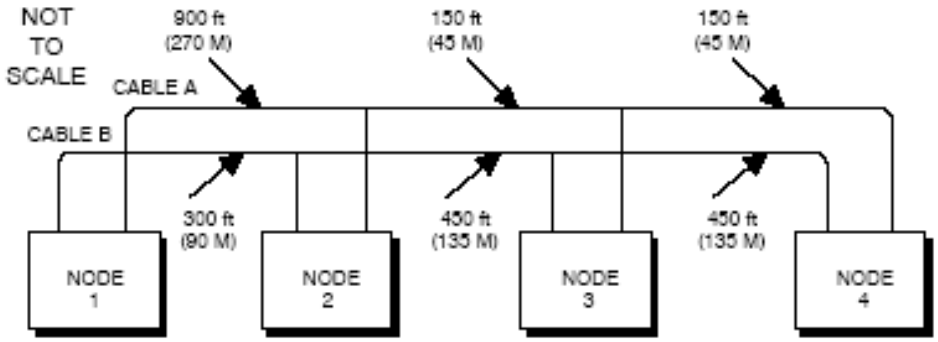


اتصال Node ها به کابل اصلی شبکه توسط Tap صورت میگیرد. وسایل Tap دارای ترمینیتور داخلی هستند و با استفاده از Jumper در مدار قرار میگیرند یا از مدار خارج میشوند. بدیهی است بجز در ابتدا و انتهای باس این ترمینیتور ها نباید در مدار قرار گیرند.

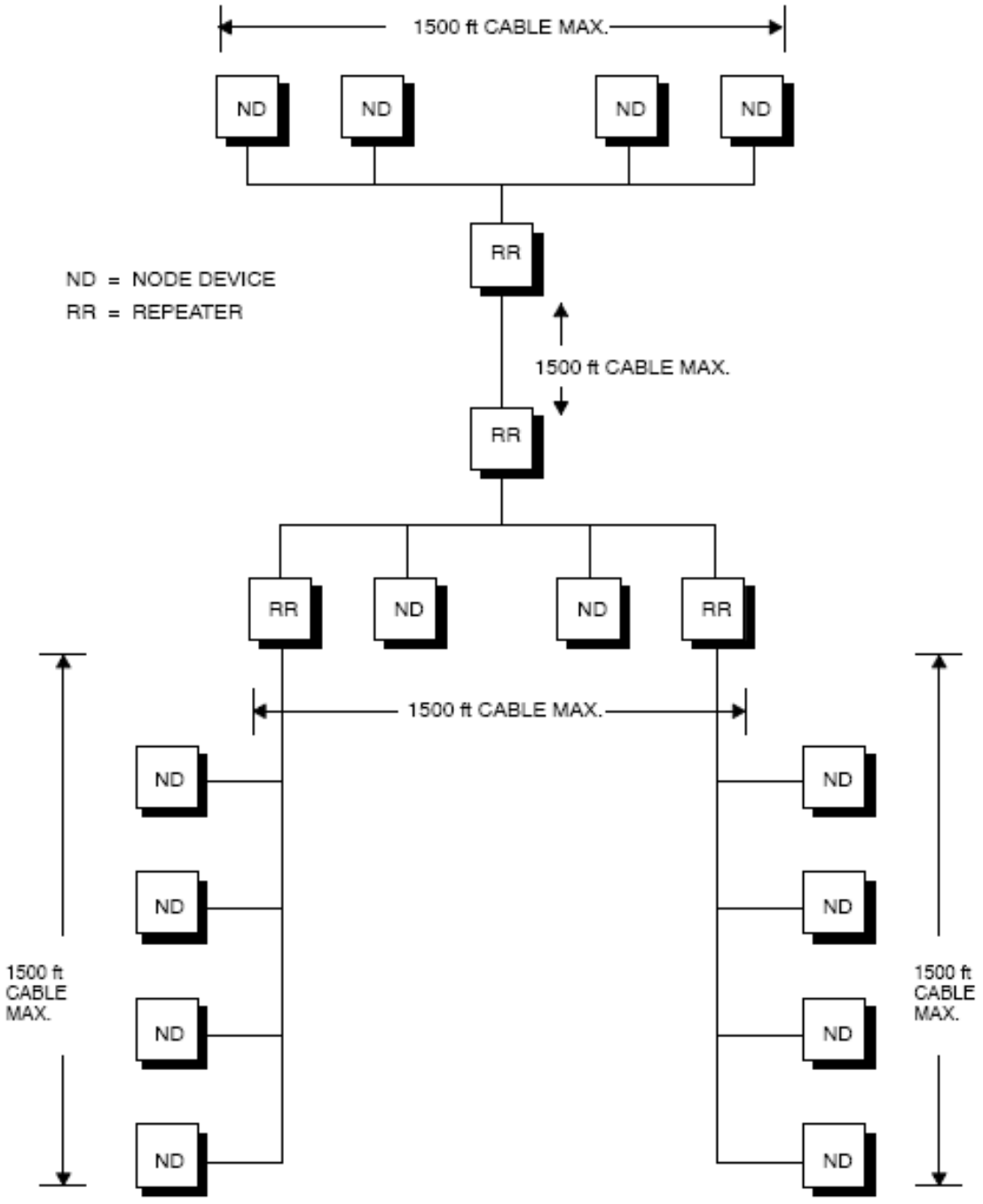


هر Tap درای محلی برای اتصال به زمین است حتی اگر وسیله ای به Tap متصل نباشد لازم است این اتصال زمین از طریق کابل Drop برقرار گردد. کابل Drop توسط کانکتور ۹ پین D-sub به وسیله متصل میگردد.

در برخی کاربردها برای قابلیت اطمینان بالاتر از کابل دوبل و اتصال دوبل برای شبکه Modbus Plus استفاده میشود. در این شبکه کابل ها با اسامی A و B شناخته میشوند در اینجا نیز طول کابل همان محدودیتهای ذکر شده را دارد به اضافه اینکه اختلاف طول کابلها بین هر دو Node مجاور نباید از ۱۵۰ متر بیشتر باشد. شکل زیر این مشکل را برای کابل ارتباطی بین Node های ۱و ۲ نشان میدهد.

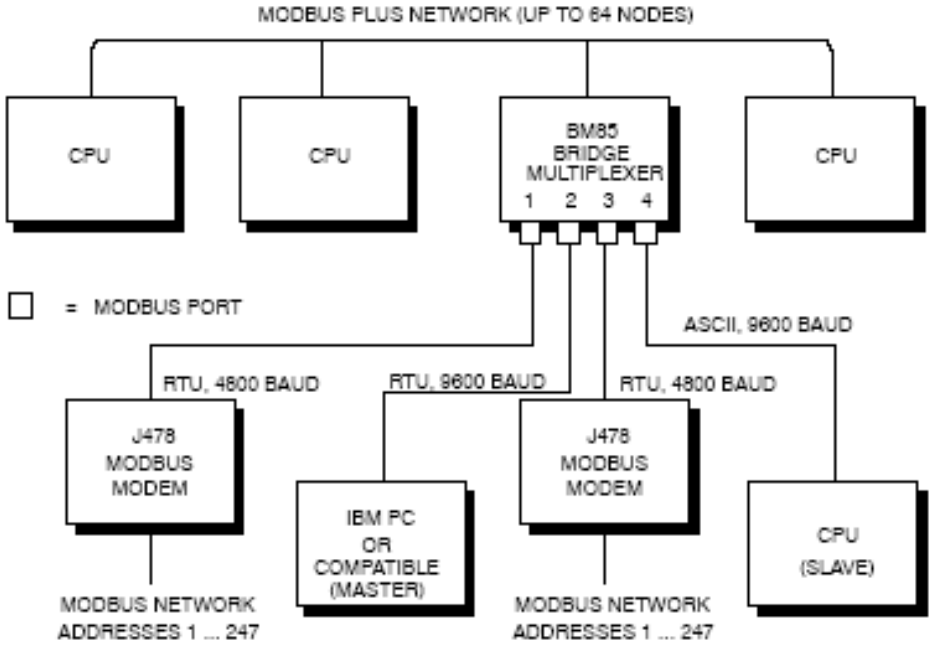


با استفاده از ریپیتر می توان شبکه را با توپولوژی های دیگری غیر از باس مانند Tree و Star نیز آرایش داد . شکل بعد نمونه ای از این توپولوژی ها را نشان می دهد. محدودیت های ذکر شده قبلی برای این ساختار نیز وجود دارد



ارتباط Modbus Plus با وسایل سریال از طریق Bridge

با استفاده از Bridge Multiplexer یا اصطلاحاً BM میتوان وسایل سریال را به شبکه Modbus Plus متصل کرد.هر مالتی پلکسر دارای ۴ پورت برای ارتباط این وسایل است که هر پورت میتواند بطور دلخواه پیکر بندی شود. Master یا Slave بودن ، انتخاب مد ASCII یا RTU ، نرخ تبادل دیتا و امثال آن قابل پیکربندی است.



در این پیکر بندی هر Master که به پورت سریال BM متصل شده باشد امکان دسترسی به سایر Master یا Slave هایی که به پورت های سریال دیگر متصل هستند را داراست. بعنوان مثال هر کدام از Master های متصل به پورت ۱ یا ۲ وسیله BM در شکل فوق میتواند به موارد زیر دسترسی داشته باشد:

- به هر کنترلر روی کل شبکه Modbus Plus
- به هر وسیله Slave متصل به شبکه مانند هر کدام از ۲۴۷ Slave متصل شده به پورت ۳
- به کنترلر Slave متصل به پورت ۴

همانطور که در شکل مشخص است هر دو شبکه متصل به پورت های ۱ و ۳ میتوانند دارای آدرس Node های مشابه باشد. بعلت نقش Bridge این آدرس ها مجزا تلقی شده و می توانند با یکدیگر به تبادل دیتا بپردازند.

توجه : نکاتی مرتبط با نصب Modbus Plus در فایل Modbus Plus Network ص ۱۵۱ به بعد هست

خلاصه تفاوتهای Modbus و Modbus Plus در جدول زیر آمده است:

Modbus	Modbus Plus
Modbus is a Master/Slave serial communications command language and protocol.	Modbus Plus is a token passing LAN containing Modbus messages as the data portion of the Modbus Plus messages.
Modbus is an open standard.	Modbus Plus is a proprietary protocol. It may

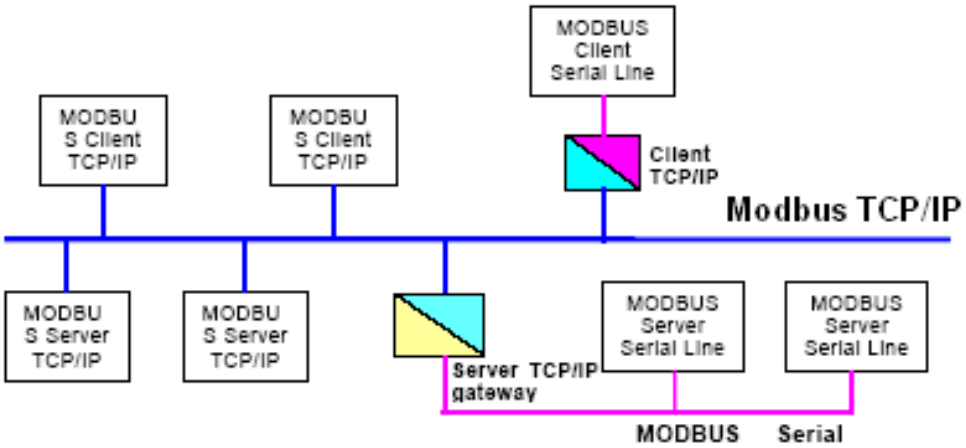
	not be used without permission of Schneider Automation.
The physical medium for Modbus is defined as RS۲۳۲ standard. However the protocol can be used in a variety of media, including phone or leased lines, radio transmissions and other networks.	The physical medium for Modbus Plus is typically a twisted pair cable or optical fiber.
Modbus is a point to point protocol.	Modbus Plus is a networked protocol.
Modbus networks are Master/Slave networks.	Modbus Plus networks are peer to peer networks.

Type of Network	
ASCII/RTU	Device Bus
ModbusPlus	Control Bus
Physical Media	Shielded twisted pairs in one shielded cable
Network Topology	Bus, tree, star with drops
Maximum Devices	
ASCII/RTU	One to one communications
ModbusPlus	۳۲ (up to ۶۴ with repeater)
Maximum Distance	
ASCII/RTU	۳۵۰m
ModbusPlus	۱۵۰۰m (۶۰۰۰m with repeaters)
(up to ۳ repeaters may be used) (min. ۱m between devices)	
Communication Methods	
ASCII/RTU	Master-Slave Query-Response Cycle (LRC error checking for ASCII) (CRC error checking for RTU)
ModbusPlus	Peer to Peer (Token passing logical ring)
Primary usage	
ASCII/RTU	Serial Communications for PLC, Variable Speed Drives, Control Systems, etc. linking MODBUS and/or RS۲۳۲/RS۴۸۵ devices in a peer-to-peer network
ModbusPlus	

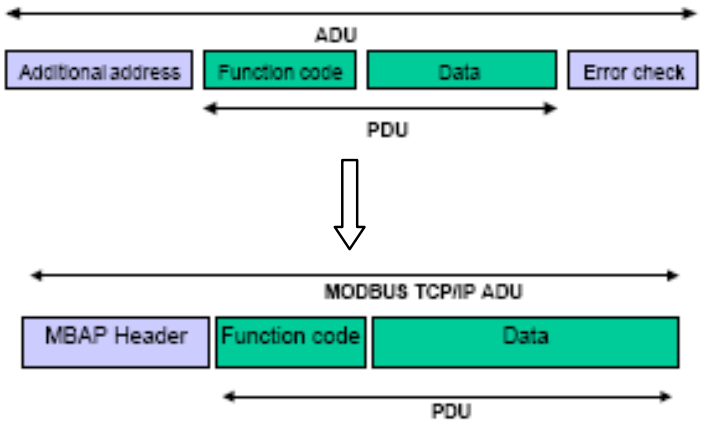
Modbus TCP/IP ٤-٤

Modbus TCP/IP که Modbus TCP هم خوانده می شود عضو دیگری از خانواده Modbus است که در سال ۱۹۹۰ توسط Modin (که امروزه زیر مجموعه Schneider Automation است) ارائه گردید و جایگاه

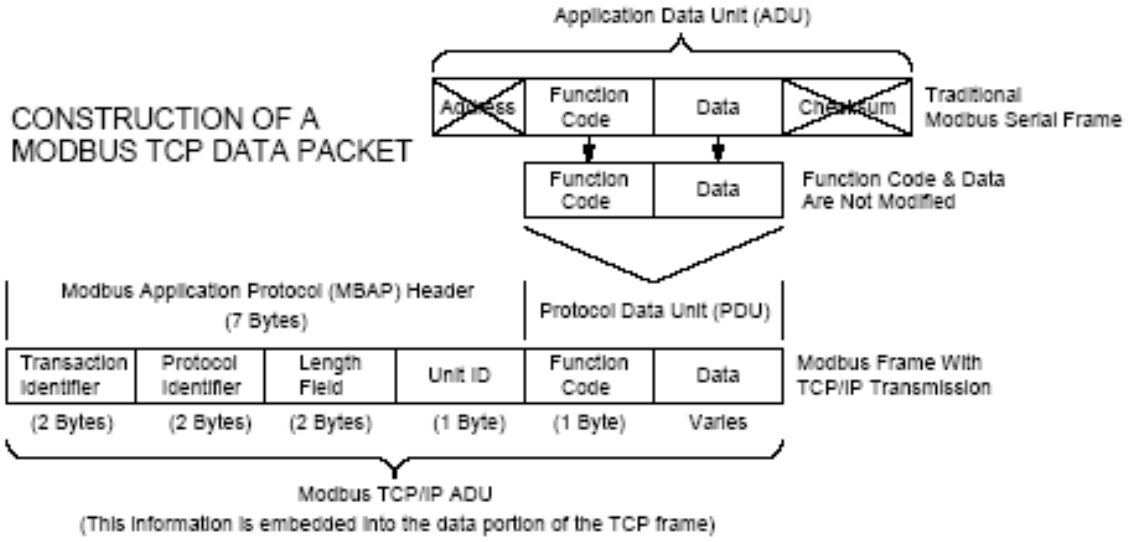
خود را در سطوح Supervision و Control از هرم اتوماسیون باز کرد. PLC ها و کامپیوترها و برخی وسایل I/O نیز توسط این شبکه با یکدیگر مرتبط شدند.



در نگاه کلی Modbus TCP/IP همان پروتکل Modbus RTU میباشد که با اینترفیس TCP/IP روی شبکه Ethernet کار میکند و در آن TCP برای اطمینان از ارسال درست دیتا و IP برای اطمینان از آدرس دهی و مسیر یابی صحیح بکار میرود. تفسیر دیتا و شناسایی اصل پیام وظیفه پروتکل Modbus است .در عمل Modbus TCP/IP فریم استاندارد پیام Modbus را به فریم استاندارد TCP که اصطلاحاً به آن فریم MBAP مخفف ModbBus Application Protocol میگویند مطابق شکل زیر تبدیل میکند :



شکل دقیقتر در صفحه بعد آورده شده است.



همانطور که ملاحظه می شود فیلدهای Address و Error Checking از فریم پایه Modbus حذف شده و بقیه فیلدها یعنی Function Code و Data در فریم TCP در کنار فیلد های جدیدی قرار میگیرند. فیلدهایی که در MBAP وجود دارند عبارتند از :

Transaction Identifier : این فیلد برای Transaction Pairing بکار میرودیعنی ارسال چند پیام روی یک ارتباط TCP توسط Client بدون اینکه منتظر پاسخ پیامهای اولیه از طرف Server باشد. در Modbus سریال Client پس از ارسال پیام باید منتظر تایید Server میماند سپس پیام بعدی را ارسال میکرد ولی در Modbus TCP/IP چندین پیام میتواند توسط Client شود بدون اینکه زمانی را برای انتظار پاسخ پیام قبلی تلف کند. تعداد اینگونه پیام ها بستگی به قابلیت و ظرفیت Server دارد. و میتواند بین ۱ تا ۱۶ باشد.

Protocol Identifier : این فیلد برای Modbus مقدار ۰ را دارد سایر مقادیر برای آینده رزرو شده اند.

Length :این فیلد تعداد بایت های موجود در فیلدهای بعد از خود یعنی سه فیلد Unit ID و Function Code و Data را نشان می دهد.

Unit Identifier : این فیلدبرای Routing است و وقتی استفاده میشود که پیامی از روی Modbus TCP لازم باشد به وسیله که روی Modbus سریال یا Modbus Plus متصل و از طریق Bridge یا Gateway با Modbus TCP ارتباط دارد ارسال شود. در اینحالت آدرس IP موجود در پیام مربوط به Gateway یا Bridge است که کار انتقال پیام به سمت دیگر را انجام میدهند.

در حالتی که وسیله مقصد روی TCP/IP قرار دارد مقدار این فیلد FF هگز است.

فانکشن کدهایی که در ADU قرار میگیرند همان مواردی هستند که در Modbus RTU به آنها اشاره شد با این وجود بمنظور آشنایی خواننده محترم با مقادیر فیلد های MBAP نمونه ای ذکر میشود. فرض کنید توسط فانکشن کد ۰۱ که برای خواندن خروجی هاست Client بخواهد خروجی های ۳-۰ را از Server بخواند در اینحالت مقادیر بسته ADU برای درخواست و نیز برای پاسخ در جداول زیر ارائه شده اند:

Modbus Request ADU Example - Read Coil Status Query

Field Name	Example Decimal (Hexadecimal)
Function Code	۱ (۰۱)
Starting Address High Order	۰ (۰۰)
Starting Address Low Order	۰ (۰۰)
Number Of Points High Order	۰ (۰۰)
Number Of Points Low Order	۴ (۰۴)

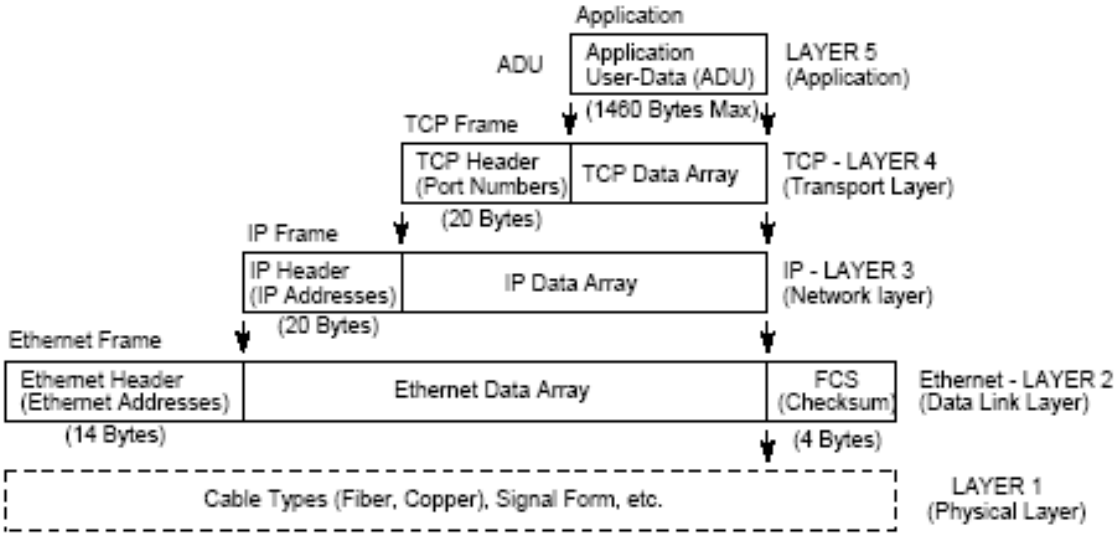
Modbus Response ADU Example Header

MBAP Header Fields	Example Decimal (Hexadecimal)
Transaction ID High Order	۰ (۰۰) *Echoed back, no change*
Transaction ID Low Order	۱ (۰۱) *Echoed back, no change*
Protocol Identifier High Order	۰ (۰۰) *Echoed back, no change*
Protocol Identifier Low Order	۰ (۰۰) *Echoed back, no change*
Length High Order	۰ (۰۰) *Server calculates*
Length Low Order	۴ (۰۴) *Server calculates.*
Unit Identifier	۲۵۵ (FF) or ۰ (۰۰) *No change*

Modbus Response ADU Example - Read Coil Status Response

Field Name	Example Decimal (Hexadecimal)
Function Code	۱ (۰۱)
Byte Count	۱ (۰۱)
Data (Coils ۳-۰)	۱۰ (۰A)

آنچه تاکنون گفته شد مربوط به فریم پیام یعنی ADU بود این پیام برای ارسال به بالاترین لایه در مدل TCP/IP تحویل داده میشود سپس مرحله به مرحله از لایه های پایین تر میگذرد تا به لایه فیزیکی برسد. شکل بعد این موضوع را بهتر نشان میدهد:



برخی نکات مربوط به Modbus TCP/IP

- بدلیل استفاده از اترنت در لایه فیزیکی دارای سرعت بالاست.
- تطابق و سازگاری آن با وسایل و سخت افزار ها بدلیل کاربرد فراوان TCP/IP زیاد است .
- تعداد وسایل متصل به شبکه بسیار زیاد است و میتواند تا حدود ۱۰,۰۰۰ وسیله را پوشش دهد.
- پیکر بندی آن ساده است اضافه کردن وسیله جدید به این شبکه پیچیدگی خاصی ندارد.