پروتکل ارتباطی خانواده PAC

برای ارتباط PLCهای خانوادهی PAC با یکدیگر و نیز با PC، از طریق شبکهی RS-485، یک پروتکل ارتباطی تعریف شده است.

در شبکهی RS-485 تمام تبادل دادهها در شبکه، توسط Master انجام میشود. در این شبکه تنها یک ماژول Master میتواند وجود داشته باشد. در صورتی که ارتباط بین PC و یک PLC برقرار شود، PC به عنوان Master شناخته میشود.

ID دستگاهها در این شبکه یک بایت بوده و بنابراین در هر شبکهی RS-485 میتوان حداکثر ۲۵۵ PLC را با یکدیگر شبکه کرد. در صورتیکه بخواهیم از طریق PC با یک دستگاه ارتباط برقرار کنیم، PC به عنوان Master شناخته شده و ID آن همیشه برابر صفر است.

برای تبادل اطلاعات در شبکه، Master اقدام به ارسال و دریافت اطلاعات بین PLCهای Slave میکند. دادهها در شبکه به وسیلهٔ فریم (Frame)هایی که در شبکه ارسال میشوند ، انتقال مییابند. این فریمها دارای یک فرم استاندارد میباشند که این فرم استاندارد همان پروتکل ارتباطی شبکه است. در اینجا ساختار فریمهای مختلفی که در یک شبکه ایجاد میشوند، توضیح داده شدهاست.

در این سیستم ۳ نوع فریم متفاوت توسط ماژولهای موجود در شبکه ایجاد میشود :

فریم Read

فریم Write

فریم Acknowledge

فریم Read :

هنگامی‌که Master بخواهد داده‌ای را از حافظه‌ی یک ماژول بخواند (Read) و آن را در محلی از حافظه‌ی خودش ذخیره نماید، عبارت زیر را بر روی شبکه می‌فرستد :

Byte	Byte	Byte	Byte	Word		Byte	Word		Byte	Word	
ID مقصد	ID مبدأ	CMD	نوع حافظه مقصد	آدرس حافظه مقصد		نوع حافظه مبدأ	آدرس حافظه مبدأ		طول داده	CRC	
				L	H		L	H		H	L

در این فریم، منظور از مبدأ، ماژول Master (یا PC) و منظور از مقصد، ماژول Slave می‌باشد که قرار است داده از حافظه‌ی آن خوانده شود. قسمت‌های مختلف این فریم به شرح زیر می‌باشد :

ID مقصد : در این بایت، شماره‌ی ID دستگاهی که قرار است داده از حافظه‌ی آن خوانده شود قرار می‌گیرد. این بایت مقداری از 0x01 تا 0xFF خواهد داشت.

ID مبدأ : در این فریم که به وسیله‌ی ماژول Master ارسال می‌گردد، مبدأ، ماژول Master است؛ بنابراین در بایت دوم این فریم ID ماژول Master قرار می‌گیرد.

CMD : در این بایت نوع فریم ارسالی مشخص می‌شود. مقدار این بایت با توجه به نوع فریم ارسالی از جدول زیر استفاده می‌شود:

CMD	نوع فریم
0	Read
1	Write
2	Acknowledge

با توجه به اینکه فریم ارسالی از نوع Read می‌باشد، در بایت مربوط به CMD، مقدار 0x00 قرار می‌گیرد.

نوع حافظه‌ی مقصد : در این بایت نوع حافظه‌ی دستگاه مقصد که داده از آن خوانده می‌شود مشخص می‌گردد. در صورتی که داده از حافظـه‌ی RAM مـاژول Slave خوانـده‌شـود، مقـدار 0x00 و در صورتی که از حافظه EEprom آن خوانـده‌شـود، مقـدار 0x01 در ایـن بایـت قـرار می‌گیرد.

آدرس حافظه‌ی مقصد : در این بخش آدرس بایتی از حافظه‌ی ماژول Slave که داده‌ها از آن و بایت‌های بعد از آن خوانده می‌شود، قرار می‌گیرد. دقت داشته باشید که آدرس مذکور از نـوع Word تعریف شده و بایت با ارزش کمتر(Lower Byte) قبل از بایت با ارزش بیشتر(Higher Byte) نوشته می‌شود.

نوع حافظه‌ی مبدأ : در ایـن بایـت، نـوع حافظـه‌ی مبـدأ (Master) کـه داده در آن نوشـته خواهدشد، تعیین می‌گردد. در صورتی‌که داده در حافظه‌ی RAM ماژول Master نوشته شود، مقدار 0x00 و در صورتی که در حافظه‌ی EEprom آن نوشته شود، مقدار 0x01 در این بایـت قرار می‌گیرد.

آدرس حافظه مبدأ : در این بخش آدرس بایتی از حافظه‌ی ماژول Master که داده‌هـا در آن و بایت‌های بعد از آن نوشته می‌شوند، قرار می‌گیرد. دقت داشته باشید که آدرس مذکور از نوع Word تعریف شده و بایت با ارزش کمتر (Lower Byte) قبل از بایت با ارزش بیشتر(Higher Byte) نوشته می‌شود.

طول داده : در این بایت، تعداد بایـت‌هـایی کـه بایـد از حافظـه‌ی Slave خوانـده شـده و در حافظه‌ی Master نوشته شوند ، تعیین می‌گردد؛ به عبارت دیگر، طول داده‌ی مورد نیاز در این بایت قرار می‌گیرد. از آنجا که این بایت حداکثر مقدار ۲۵۵ را می‌تواند به خود اختصاص دهـد، پس در هر فریم حداکثر می‌توان ۲۵۵ بایت را انتقال داد.

CRC : در این بخش از فریم، CRC مربوط به دادهٔ ارسالی قرار می‌گیرد. این CRC از نوع ۱۶ بیتی بوده و همان‌طور که مشاهده می‌شود، بایت با ارزش کمتر (Lower Byte) قبل از بایت با ارزش بیشتر (Higher Byte) قرار می‌گیرد.

فریم Write :

هنگـــامی کـــه Master بخواهـــد داده‌ای را در محلـــی از حافظـــه‌ی یـــک مـــاژول Slave بنویسد(Write)، عبارت زیر را بر روی شبکه می‌فرستد :

Byte	Byte	Byte	Byte	Word		Byte	...	Word	
ID مقصد	ID مبدأ	CMD	نوع حافظه مقصد	آدرس حافظه مقصد		طول داده	داده	CRC	
				L	H			H	L

در این فریم، منظور از مبدأ، ماژول Master (یا PC) و منظور از مقصد، ماژول Slave می‌باشد که قرار است داده در حافظه‌ی آن نوشته شود. قسمت‌های مختلف این فریم به شرح زیر می‌باشند:

ID مقصد : در این بایت، شماره‌ی ID دستگاهی که قرار است داده در حافظه‌ی آن نوشته شود قرار می‌گیرد. این بایت مقداری از 0x01 تا 0xFF خواهد داشت.

ID مبدأ : در این فریم که به وسیله‌ی ماژول Master ارسال می‌شود، مبدأ، ماژول Master است؛ بنابراین در بایت دوم این فریم، ID ماژول Master قرار می‌گیرد.

CMD : در این بایت نوع فریم ارسالی مشخص می‌شود. مقدار این بایت با توجه به نوع فریم ارسالی طبق جدول زیر می‌باشد :

CMD	نوع فریم
0	Read
1	Write
2	Acknowledge

با توجه به اینکه فریم ارسالی از نوع Write می‌باشد، در بایت مربوط به CMD، مقدار 0x01 قرار می‌گیرد.

نوع حافظه‌ی مقصد : در این بایت نوع حافظه‌ی PLC مقصد، که داده در آن نوشته می‌شود، مشخص می‌گردد. در صورتی که داده در حافظه‌ی RAM ماژول Slave نوشته شود، مقدار 0x00 و در صورتی که داده در حافظه‌ی EEprom آن نوشته شود، مقدار 0x01 در این بایت قرار می‌گیرد.

آدرس حافظه‌ی مقصد : در این بخش آدرس بایتی از حافظه‌ی ماژول Slave که داده‌ها در آن و بایت‌های بعد از آن نوشته می‌شوند، قرار می‌گیرد. دقت داشته باشید که آدرس مذکور از نوع Word تعریف شده و بایت با ارزش کمتر(Lower Byte) قبل از بایت با ارزش بیشتر(Higher Byte) نوشته می‌شود.

طول داده : در این بایت، تعداد بایت‌هایی که باید در حافظه‌ی Slave نوشته شوند، تعیین می‌گردد؛ به عبارت دیگر، طول داده‌ی مورد نظر در این بایت قرار می‌گیرد. از آنجا که این بایت حداکثر مقدار ۲۵۵ را می‌تواند به خود اختصاص دهد، پس در هر فریم حداکثر می‌توان ۲۵۵ بایت را انتقال داد.

داده: در این بخش دادهٔ مورد نظر که حداکثر می‌تواند شامل ۲۵۵ بایت باشد، قرار می‌گیرد.

CRC : در این بخش از فریم، CRC مربوط به داده‌ی ارسالی قرار می‌گیرد. این CRC از نوع ۱۶ بیتی بوده و همان‌طور که مشاهده می‌شود، بایت با ارزش کمتر (Lower Byte) قبل از بایت با ارزش بیشتر (Higher Byte) قرار می‌گیرد.

فریم Acknowledge :

پس از آنکه ماژول Master یک فریم Write برای یک ماژول Slave بفرستد و Slave آن را به درستی دریافت کند، ماژول Slave یک فریم با عنوان «فریم Acknowledge» برای Master می‌فرستد. بدین وسیله ماژول Master متوجه می‌شود که Slave فریم ارسالی را به درستی دریافت کرده است. ساختار فریم Acknowledge به صورت زیر است :

Byte	Byte
ID مقصد	CMD

در این فریم منظور از ID مقصد، ID ماژول Master می‌باشد؛ چرا که فریم برای Master ارسال شده است؛ در بایت مربوط به CMD مقدار 0x02 قرار می‌گیرد (با توجه به جدول).

نکته : در صورتی که ماژول Master یک فریم از نوع Read برای Slave بفرستد، ماژول Slave در پاسخ، برای Master یک فریم از نوع Write ارسال می‌کند. در این فریم، مقادیر زیر در بخش‌های مختلف آن قرار می‌گیرد :

ID مقصد : در این بایت، شماره‌ی ID ماژول Master که قرار است داده در حافظه‌ی آن نوشته شود قرار می‌گیرد.

ID مبدأ : در این فریم که به وسیله‌ی یک ماژول Slave ارسال می‌گردد، مبدأ همان ماژول Slave است، بنابراین در بایت دوم این فریم، ID ماژول Slave که داده را برای Master می‌فرستد قرار می‌گیرد.

CMD : با توجه به اینکه فریم ارسالی از نوع Write می‌باشد، در بایت مربوط به CMD، مقدار 0x01 قرار می‌گیرد.

نوع حافظه‌ی مقصد : در این بایت نوع حافظه‌ی ماژول Master که داده در آن نوشته می‌شود مشخص می‌گردد. در صورتی‌که داده در حافظه‌ی RAM ماژول Master نوشته شود، مقدار 0x00 و در صورتی‌که داده در حافظه‌ی EEprom آن نوشته شود، مقدار 0x01 در این بایت قرار می‌گیرد.

آدرس حافظه‌ی مقصد : در این بخش آدرس بایتی از حافظه‌ی ماژول Master که داده‌ها در آن بایت و بایت‌های بعد از آن نوشته می‌شوند، قرار می‌گیرد. دقت داشته باشید که آدرس مذکور از نوع Word تعریف شده و بایت با ارزش کمتر(Lower Byte) قبل از بایت با ارزش بیشتر(Higher Byte) نوشته می‌شود.

طول داده : در این بایت، تعداد بایت‌هایی که باید در حافظه Master نوشته شوند، تعیین می‌گردد ؛ به عبارت دیگر، طول داده‌ی مورد نظر در این بایت قرار می‌گیرد.

داده: در این بخش داده‌ی مورد نظر که حداکثر می‌تواند شامل ۲۵۵ بایت باشد ، قرار می‌گیرد.

CRC : در این بخش از فریم، CRC مربوط به داده‌ی ارسالی قرار می‌گیرد. این CRC از نوع ۱۶ بیتی بوده و همان‌طور که مشاهده می‌شود، بایت با ارزش کمتر (Lower Byte) آن قبل از بایت با ارزش بیشتر (Higher Byte) آن قرار می‌گیرد.

نکته: در صورتی‌که بین بایت‌های ارسالی برای یک ماژول تأخیری ایجاد شود، ماژول مورد نظر مدتی که با عنوان Timeout شناخته می‌شود منتظر مانده و در صورت عدم دریافت داده‌ی دیگر، بایت‌های دریافتی را به‌عنوان یک فریم پردازش می‌کند. در صورتی‌که یک فریم به طور کامل به یک دستگاه نرسیده باشد، هیچ عملی انجام نمی‌دهد؛ و در صورتی‌که فریم را به‌طور کامل دریافت کرده باشد، عکس العمل مرتبط با آن (ارسال یک فریم Write یا Acknowledge) را انجام می‌دهد.

نکته: ماژول Master پس از ارسال یک فریم برای Slave، منتظر پاسخ آن می‌ماند. در صورتی‌که پس از مدت معینی از ماژول Slave جوابی دریافت نشود، Master عمل ارسال فریم را تکرار می‌کند. تعداد تکرارهای این ارسال به وسیله گزینه‌ای با عنوان Retry attempts مشخص می‌گردد.

مثال: ماژول Master می‌خواهد ۳ بایت را از آدرس 288 (0x120) حافظه‌ی RAM یک ماژول Slave با ID شماره ۵ بخواند. در این حالت این فریم را بر روی شبکه می‌فرستد (فرض می‌کنیم ID ماژول Master برابر ۱ است) :

0501000020010000003XXXX

این فریم به‌صورت زیر تحلیل می‌شود :

ID مقصد	ID مبدأ	CMD	نوع حافظه مقصد	آدرس حافظه مقصد		نوع حافظه مبدأ	آدرس حافظه مبدأ		طول داده	CRC	
				L	H		L	H		H	L
05	01	00	00	20	01	00	00	00	03	XX	XX

ماژول Slave پس از دریافت فریم فوق، فریم زیر را برای ماژول Master می‌فرستد:

010501000000003A1246FXXXX

ID مقصد	ID مبدأ	CMD	نوع حافظه مقصد	آدرس حافظه مقصد		طول داده	داده	CRC	
				L	H			H	L
01	05	01	00	00	00	03	A1246F	XX	XX

در فریم فوق، ماژول Slave، با ارسال یک فریم Write به ماژول Master پاسخ می‌دهد. همان‌طور که مشاهده می‌شود، داده‌ای که در اینجا برای ماژول Master فرستاده می‌شود، یعنی همان داده‌ای که در بایت‌های ۲۸۸، ۲۸۹ و ۲۹۰ ماژول Slave وجود داشته است، برابر 0xA1246F می‌باشد.

آشنایی با CRC :

CRC ، ابزاری است که برای اطمینان از صحت اطلاعات مبادله شده در شبکه از آن استفاده می‌شود. در انتهای هر فریم ارسالی در شبکه، اطلاعاتی به صورت کد با عنوان CRC قرار می‌گیرد که ماژول گیرنده‌ی فریم، با بررسی آن متوجه می‌شود که اطلاعاتی که دریافت کرده به درستی و کامل دریافت شده یا اینکه در انتقال اطلاعات مشکلی وجود داشته است.

CRCها با استفاده از داده‌های موجود در فریم ایجاد می‌شوند؛ به این معنی که دستگاه فرستنده‌ی فریم، پس از تعیین داده‌های موجود در فریم، CRC مربوط به آنها را طبق یک الگوریتم مشخص ایجاد می‌کند و در انتهای فریم به همراه سایر داده‌های فریم برای دستگاه گیرنده می‌فرستد. در ماژول گیرنده، طبق همان الگوریتم، CRC داده‌های موجود در فریم ایجاد می‌شود و با CRC موجود در انتهای فریم دریافتی مقایسه می‌گردد. درصورتی‌که این دو با هم برابر باشند، نشانه‌ی آن است که داده‌ها به درستی دریافت شده‌اند و اگر مغایر باشند، یعنی داده‌ها به درستی دریافت نشده‌اند و اشکالی در ارسال یا دریافت وجود داشته است.

CRCها، کدهای استانداردی هستند که به صورت ۸ بیتی، ۱۶ بیتی و یا ۳۲ بیتی ایجاد می‌شوند. برای تولید CRCها الگوریتم‌های مختلفی وجود دارد. برای مثال بایت‌های اطلاعات فریم را با یکدیگر XOR می‌کنند و حاصل را به عنوان CRC داده‌های موجود در فریم می‌شناسند.

در این سیستم از CRC ۱۶ بیتی استفاده می‌شود. الگوریتم این سیستم برای ایجاد CRC، بر مبنای استفاده از یک چندجمله‌ای قرار دارد.

برنامه‌ی نوشته‌شده به زبان C و جدول مربوطه، جهت ایجاد CRC مربوط به داده‌های موجود در یک فریم در این سیستم در ادامه نشان داده شده‌است.

```
U16 CRC16_Ram(U16 Length, U8 *Pointer)
{
     U16  CRC= 0xFFFF;
     U16  i;

     for  (i=  0;  i  <  Length;  i++)  CRC  =  (CRC  <<  8)  ^
CrcTable[(CRC >> 8) ^ Pointer[i]];
     return(CRC);
}
```

در این برنامه، متغیر Length طول داده‌ای را که باید CRC آن محاسبه شود، بر حسب تعـداد بایت نشان می‌دهد. همچنین متغیر Pointer[i] ، اشاره‌گری است که در هر لحظه به بایـت iام داده‌ای که باید CRC آن را محاسبه نمود، اشاره می‌کند. جدول CRC برنامـه بـه صـورت زیـر است:

```
CrcTable[256]=
     {
          0X0000, 0XC1C0, 0X81C1, 0X4001, 0X01C3, 0XC003, 0X8002, 0X41C2,
          0X01C6, 0XC006, 0X8007, 0X41C7, 0X0005, 0XC1C5, 0X81C4, 0X4004,
          0X01CC, 0XC00C, 0X800D, 0X41CD, 0X000F, 0XC1CF, 0X81CE, 0X400E,
          0X000A, 0XC1CA, 0X81CB, 0X400B, 0X01C9, 0XC009, 0X8008, 0X41C8,
          0X01D8, 0XC018, 0X8019, 0X41D9, 0X001B, 0XC1DB, 0X81DA, 0X401A,
          0X001E, 0XC1DE, 0X81DF, 0X401F, 0X01DD, 0XC01D, 0X801C, 0X41DC,
          0X0014, 0XC1D4, 0X81D5, 0X4015, 0X01D7, 0XC017, 0X8016, 0X41D6,
          0X01D2, 0XC012, 0X8013, 0X41D3, 0X0011, 0XC1D1, 0X81D0, 0X4010,
          0X01F0, 0XC030, 0X8031, 0X41F1, 0X0033, 0XC1F3, 0X81F2, 0X4032,
          0X0036, 0XC1F6, 0X81F7, 0X4037, 0X01F5, 0XC035, 0X8034, 0X41F4,
          0X003C, 0XC1FC, 0X81FD, 0X403D, 0X01FF, 0XC03F, 0X803E, 0X41FE,
          0X01FA, 0XC03A, 0X803B, 0X41FB, 0X0039, 0XC1F9, 0X81F8, 0X4038,
          0X0028, 0XC1E8, 0X81E9, 0X4029, 0X01EB, 0XC02B, 0X802A, 0X41EA,
          0X01EE, 0XC02E, 0X802F, 0X41EF, 0X002D, 0XC1ED, 0X81EC, 0X402C,
          0X01E4, 0XC024, 0X8025, 0X41E5, 0X0027, 0XC1E7, 0X81E6, 0X4026,
          0X0022, 0XC1E2, 0X81E3, 0X4023, 0X01E1, 0XC021, 0X8020, 0X41E0,
          0X01A0, 0XC060, 0X8061, 0X41A1, 0X0063, 0XC1A3, 0X81A2, 0X4062,
          0X0066, 0XC1A6, 0X81A7, 0X4067, 0X01A5, 0XC065, 0X8064, 0X41A4,
          0X006C, 0XC1AC, 0X81AD, 0X406D, 0X01AF, 0XC06F, 0X806E, 0X41AE,
          0X01AA, 0XC06A, 0X806B, 0X41AB, 0X0069, 0XC1A9, 0X81A8, 0X4068,
          0X0078, 0XC1B8, 0X81B9, 0X4079, 0X01BB, 0XC07B, 0X807A, 0X41BA,
          0X01BE, 0XC07E, 0X807F, 0X41BF, 0X007D, 0XC1BD, 0X81BC, 0X407C,
          0X01B4, 0XC074, 0X8075, 0X41B5, 0X0077, 0XC1B7, 0X81B6, 0X4076,
          0X0072, 0XC1B2, 0X81B3, 0X4073, 0X01B1, 0XC071, 0X8070, 0X41B0,
          0X0050, 0XC190, 0X8191, 0X4051, 0X0193, 0XC053, 0X8052, 0X4192,
          0X0196, 0XC056, 0X8057, 0X4197, 0X0055, 0XC195, 0X8194, 0X4054,
          0X019C, 0XC05C, 0X805D, 0X419D, 0X005F, 0XC19F, 0X819E, 0X405E,
          0X005A, 0XC19A, 0X819B, 0X405B, 0X0199, 0XC059, 0X8058, 0X4198,
          0X0188, 0XC048, 0X8049, 0X4189, 0X004B, 0XC18B, 0X818A, 0X404A,
          0X004E, 0XC18E, 0X818F, 0X404F, 0X018D, 0XC04D, 0X804C, 0X418C,
          0X0044, 0XC184, 0X8185, 0X4045, 0X0187, 0XC047, 0X8046, 0X4186,
          0X0182, 0XC042, 0X8043, 0X4183, 0X0041, 0XC181, 0X8180, 0X4040
     };
```