江端さんのDIY奮闘記 EtherCATでホームセキュリティシステムを作る（3）：

「老人ホーム 4.0」がやって来る

http://eetimes.jp/ee/articles/1506/22/news017.html

私がEtherCATでホームセキュリティシステムを構築しなければならない切実な理由――それは「介護」です。自分の身を自分で介護する。そんな次世代の介護システム実現のために、EtherCATを利用できると考えたのです。後半では、PCベースでEtherCATスレーブを作る方法をご紹介します。

2015年06月22日 09時45分 更新

[江端智一, EE Times Japan]

FA（ファクトリオートメーション）を支える「EtherCAT」。この超高度なネットワークを、無謀にも個人の“ホームセキュリティシステム”向けに応用するプロジェクトに挑みます……!!⇒「江端さんのDIY奮闘記 EtherCATでホームセキュリティシステムを作る」連載一覧

―― 掃除機、洗濯機、炊飯器……。

戦後、これらの機器によって家事は一貫してより楽に、より手を抜いて済ませる方向を目指して歩んで来たと言えよう。

しかし、家庭内にはそれら文明の光が充分に届いていない分野が残されていた。

21世紀の現在においても、省力化が行き届いていない未開の分野が存在するのである。

それが育児分野である。

育児界における進歩は、たかだか紙おむつや粉ミルク程度に留まっており、その進歩は全自動洗濯機や自動食器洗い乾燥機が達成したレベルと比較して著しく低いと言わざるを得ない。

かかる現状を鑑（かんが）みたとき、電機メーカーが開発を急ぐべき商品はもはや言うまでもないだろう。

「全自動育児機」

である。

それは未開拓の沃野（よくや）。
先の見えない景気の低迷とヒット商品の不在に悩むメーカーに差す、一条の光。

一気に世界のシェアを独占できるまたとないチャンス。
育児界のパラダイムシフト。
人類にとっての新たなステージの幕開け。
シーモンキーの飼育並みのお手軽さ。
たった1つの「育児これっきりボタン」ですくすく育ちます。

後世、人々は言うだろう。
人類の偉大な発明は、火、ネジ、車輪そして全自動育児機だと。

とはいえ、この商品にも欠点がないわけではない。

少子化という現在のトレンドである。
少子化が加速している現在、この商品の将来性にやや難があるのは事実だろう。

だが心配は無用である。

全自動育児機の開発技術は、もう1つの優れた商品の開発にも応用できるからである。

その商品とは

「全自動寝たきり老人介護機」

である。

現在、先進国と言われる国々では、どこも国民の高齢化が悩みの種である。育児分野が先細りなのに対し、老人分野は今後の成長が大いに期待できよう。介護に要する精神的肉体的負荷を大幅に軽減できるこの商品は、喝采をもって世に容れられるであろう。

まさに「ゆりかごから墓場まで」

これからの電機メーカーの進むべき、新たな指標がここにある。

世間の非難は浴びるにしても。

冒頭から過激な文章を掲載しましたが、これは、十数年前、育児で疲れ果てていた私たち夫婦に、私のコラムの師匠である先輩が贈ってくれた一通のメールです（師匠、あの時は、ありがとうございました。夫婦で爆笑させていただきました）。

さて、当時の私たち夫婦にとって、「全自動寝たきり老人介護機」の方はともかくとして、「全自動育児機」は、心底欲しいと思いました。

「全自動育児機」とまでは言わないまでも、『世の中の全ての育児に関わる保護者に、毎日、連続4時間の睡眠を与えることさえできれば、育児ノイローゼによる悲劇的な事件は劇的に減る』と思っていましたし、今でもそう信じています。

そして十数年を経て、子どもたちの手が離れるようになってきた今、私たち夫婦は、「全自動寝たきり老人介護機」の話を、単なるシャレや冗談として話すことができない段階に突入しているのです。

「老人ホーム4.0」の時代へ

先月のコラムで、私は「今の私には、大量のDIO（Digital Input/Output）をわが家の至る所に準備する、切実な理由があります」と記載しました。

その切実な理由とは「介護」です。

私の父と母は、ここ数年の間で、2人そろって、あれよ、あれよと言う間に、「歩けなく」なり、「食べられなく」なり、「思い出すことができなく」なりました。その間の出来事については割愛しますが、この「人が壊れていくプロセス」は、姉、嫁、子どもたち、そして私の体と心を、今なお破壊し続けています。

―― 時間軸方向に希望がない

今の父と母は、確定した未来の私です。

この問題を、人工知能や、介護ロボットや……（えっと、もう出てこない。他に何かあるんだっけ？）などで解決しようとする動きがありますが、私は難しいと思うのです。現在の介護現場が要求している仕様を満足できない上に、コストが見合わない ―― 人間の人件費に対して、ロボットの運用コストは驚くほど高い ―― からです。

加えて、介護問題を技術面からアプローチしている研究者たちは、自分の都合のよい介護の現場を想定して、フリーダムに好きな研究をやって、無意味な論文を量産しています。

例えば、『自分が飲んでいるクスリの数さえ認定できない認知症の人（例えば私の父）に、ウェアブルデバイスを装着させる』などという、あり得ない介護環境を「創り出し」ます。

そして、その環境下で、（使われるかどうか分からない）適当なシステムや装置を考案し製作し、恐ろしく少ない被験者から、山のようなデータを量産します。ゴチャゴチャとデータが記載されたその論文には、その装置を実現するために必要な市場規模や製品コスト試算すらないのです。

さらに、政府は、この程度の研究に対して、ろくな適用先も、実用化の目星もつけないまま、研究補助金をばらまいています。

―― お前ら、介護の現場、ナメてんのか

と、叫びたくなる衝動にかられます*)。

*)全ての介護研究が、このような「ナメた研究」というわけではありません(念のため)。

私は、この「希望のない確定した未来」から逃れられないまでも、せめて一矢報いたいと思うのです。

この連載で、私の提唱するコンセプトは、「Industrie4.0」ならぬ、

「老人ホーム4.0」

です。

Industrie4.0やIIC(Industrial Internet Consortium)で、IoTが、これからどう発展しようが、そんなことは私の知ったことではありません。

私のターゲットはあくまで「私」であり「ホーム」です。そして、ここで言う「老人ホーム」とは、私(老人)の自宅(ホーム)のことです。

これまで「機械でホームを守る」という意味で使われてきた「ホームセキュリティ」という用語を、「機械に(ホームの中の)私を守らせる」という意味にパラダイムシフトさせます。

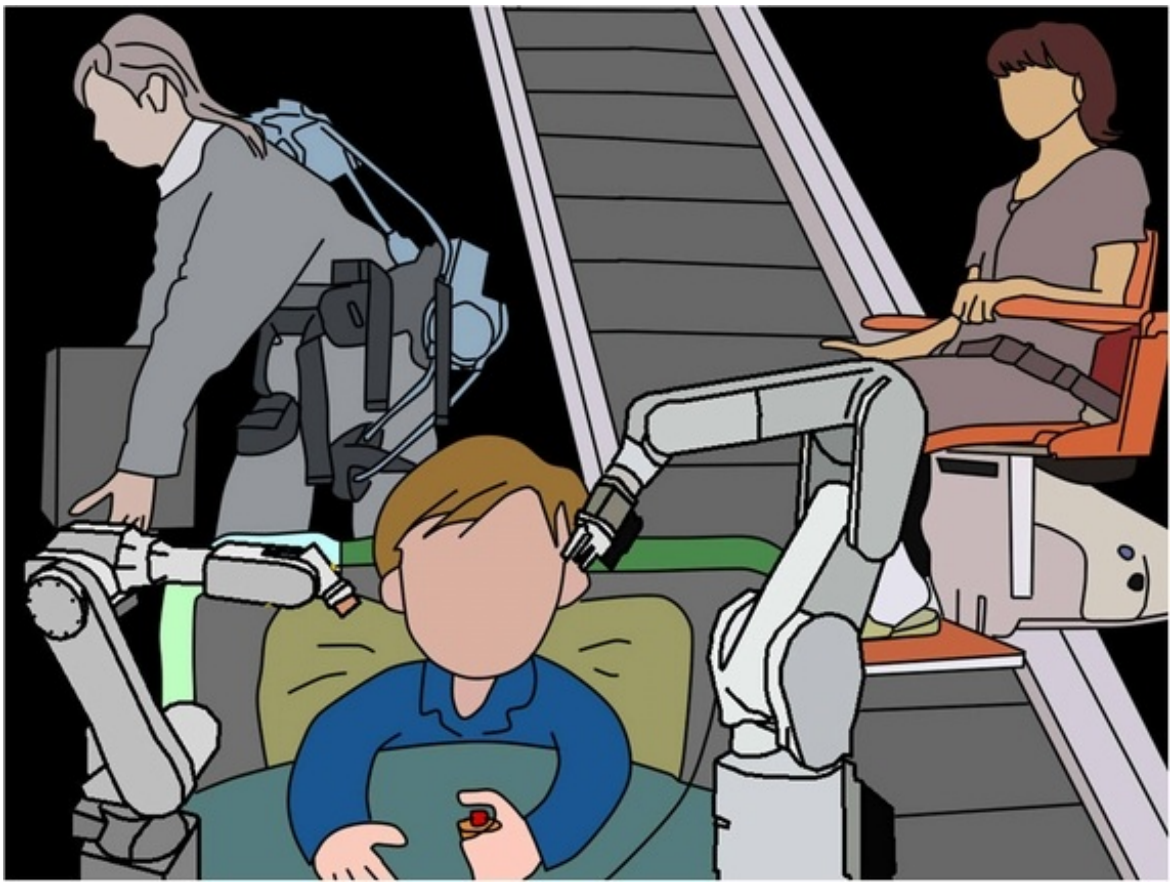
私は、ここに、私の老後の人生のために、私の自宅の隅々に、私のために必要な介護アシスト機器を「自分で作り、自分だけのために動かす」ホームセキュリティシステムの検討を開始します。

―― もう、私は研究者も政府も当てにしない。自分の老後という闇を、自分の技術力だけで切り開くのだ ―― そう考えた時、これ(EtherCAT)は「使える」と思ったのです。

既に私は、自宅でイーサネットケーブルを、窓から窓に渡し、庭に溝を掘って張り巡らせています(壁の中にLAN配線をするとか、コストのかかることを考えたら負けです。ケーブルは、切れたらまた張り直せばいいのです)。

あとは、メイド（EtherCATスレーブ）を設置さえすれば、家中どこにでもDIOポートがあるという環境が実現できます。

そうすると、移動椅子が私を運び、ロボットが私にご飯を食べさせ、パワードスーツが自動的に私に装着されて私を外に連れ出してくれるという、私の考える「老人ホーム4.0」のインフラは整うのです。



そのためにも、できるだけ早いとこ世間を巻き込んで、EtherCATをはやらせておかなければーー。そして、EtherCATスレーブの劇的なコストダウン（数万円→数百円）を、あと10年以内（私の退職まで）に実現しておかなければーー。

私には時間がないのです。

本を読めば目がかすみ、走れば腱を切り、歩いているだけで突然ギックリ腰が襲う。これが、「中年」というものです。

定年後に着手したのでは遅いのです。もう、私は「老人ホーム4.0」の具体的な設計を開始し、システム構築を完了させなければなりません。

私専用の介護ロボットを、自ら動かさなければならない、その日までに――。

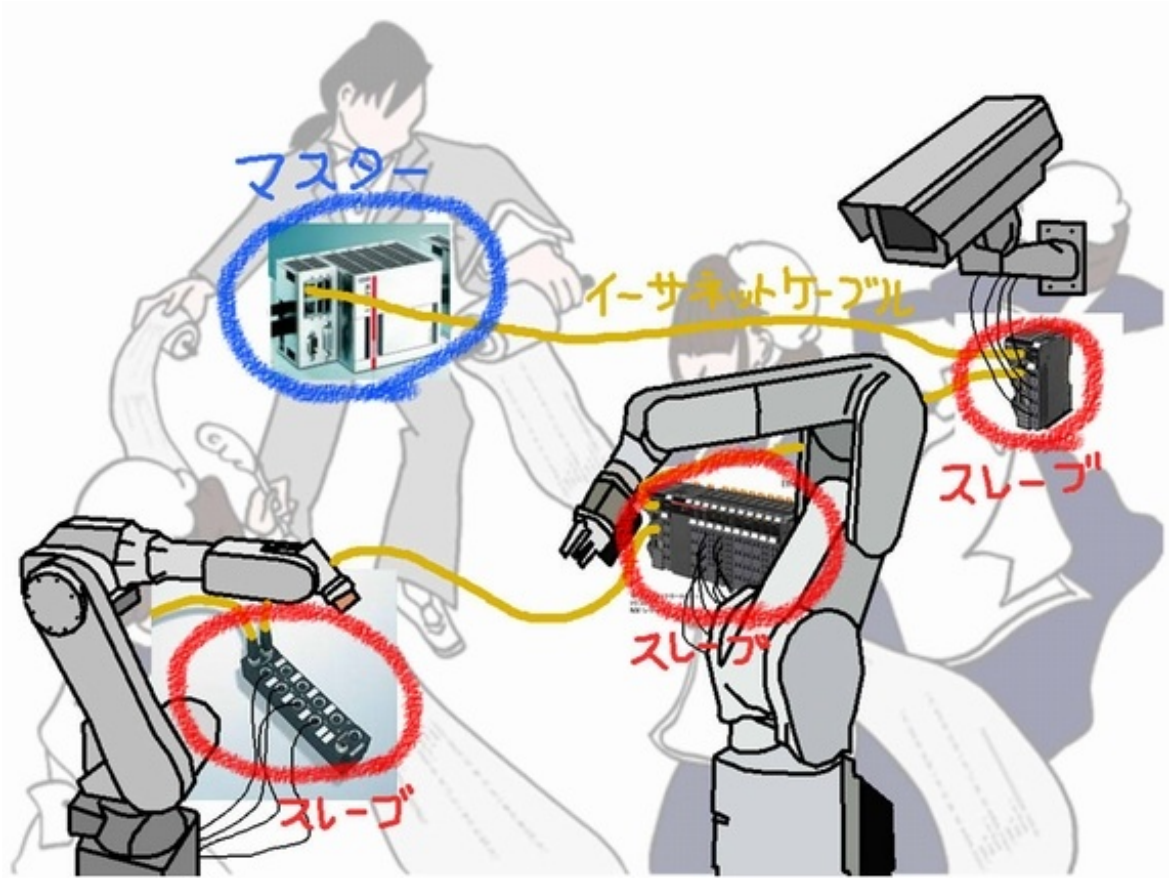
SDO通信とは？

こんにちは、江端智一です。

前回は、実際にスレーブに制御データを送って、制御システムを稼働させる通信、プロセスデータオブジェクト（Process Data Object：PDO）通信と、PCベースのEtherCATマスタSOEM（Simple Open EtherCAT Master）について説明しました。

今回は、EtherCATのもう一つの通信である、サービスデータオブジェクト（Service Data Object：SDO）通信と、PCベースのEtherCATスレーブ（light ethercat slave）についてお話したいと思います（なお、今回も、EtherCATマスタ→ご主人様、EtherCATスレーブ→メイドで押し通します）。

PDO通信は、前回ご紹介したように「ご主人様が全メイド（最大65000人）に対して、1秒間に100回以上もの指示を出し続ける」大変せわしない通信です。



　これに対して、SDO通信は、PDO通信を始める前に、『ご主人様が全てのメイドの素性を調べ上げる』、またはPDO通信の合間に『メイドたちの（健康）状態をチェックする』ことを目的とする通信です。

　何しろ、メイドたちは、1秒間に100〜1000回以上もの指示を出して、それを実行しているのですから、息切れする（制御が間に合わない、同期が外れる）ことだってあります。

　SDO通信は、マスタが1人のメイドとだけメッセージを交換することから、「メールボックス（直訳すると「郵便ポスト」）通信」とも言われています。

　私、『これ、センスのあるネーミングだなぁ』と感心しているんです。

　ツンデレなご主人様は、メイドの家の郵便ポストに、こっそりメイド個人宛ての手紙を投げ込んでおきます。メイドが、忙しい仕事（PDO通信とロボット制御の仕事）の合間に、ときどきメールボックス（郵便ポスト）を開くと、そこには、ご主人さまからの私信メールが入っているわけです。

　まるで、小学生の女の子向けのラブコメ漫画のようじゃないですか。

実際は、そんなロマンチックな話ではないんですけどね。

メイドは、メールボックス通信のメッセージに対しては、（PDO通信のように、最短125μ秒（1秒/8000回）以内というような、瞬速の対応をしなくても）都合の良いタイミングで応答してもいいーー。SDOは、そんな通信です。

そして、所定のタイムアウト時間を超えてメイドから返信が戻ってこないと、ご主人様は「メイドが（故障で）倒れた！」と判断して、所定のトラブルシューティングの準備に入ることになります。

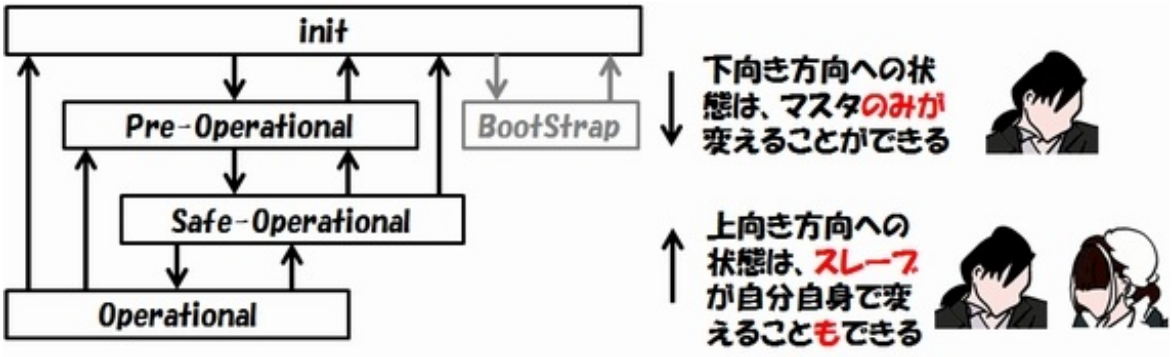
タイムアウトは、メイドが担当する機械（ロボット、センサー、スイッチ）によって、0.1秒から10秒以上までさまざまです（ずいぶん短い時間のように思われるかもしれませんが、100μ秒単位のタイミングで動作する制御システムにとって、10秒は、「永遠」と同義です）。

さて、EtherCATでは、この「メイドたちを（製造）ラインに徐々に慣れさせて行く」とか「メイドが（故障で）倒れた」などの状態（ステータス）を、以下の4段階で把握し、コントロールします。

状態	内容	一言で言うと・・・
Init	メールボックス・プロセスデータ通信不可	メイドは、電源がONになっているだけです（→なんにもできません）。
Pre-Operational	メールボックス通信可能	メイドは、「郵便ポスト」で、ご主人様と交通ができるだけです。
Safe-Operational	プロセスデータ入力通信可能	交通だけでなく、ご主人様からの命令を「聞く」ことができます。
Operational	プロセスデータ入出力通信可能	交通も、ご主人様からの命令を「聞く」ことも、ご主人様に結果を「お知らせする」こともできます（→なんでもできます）

基本的に、メイドたちの健康状態を決めるのは、ご主人様です。ご主人様が「O.K.」と言ってくれない限り、メイドは仕事に出ることができない、または、その仕事の内容を制限されることになります。

ただ、メイドは自分の状態が悪いと判断した場合に限り、ご主人様の許可を得ず、自分から「降格」を決めることができます。しかし、自分から「昇格」を申し出ることはできません。



EtherCATは、ご主人様とメイドからなる制御LANですが、厳しいサラリーマン社会の1つの姿でもあるわけです。

紙面が押してきましたので、SDOの3つの通信手段、SDOオブジェクトの内容、6万5000人のメイドを統一管理するメモリ管理方法、その他については、次回以降でお話したいと思います（なんか、この連載長くなりそうな気がしてきました）。

PCベースで「メイド」を作る

今回は、Visual C++ 2010（無償版）を使った、EtherCATマスタであるSOEM（Simple Open EtherCAT Master）のデバッグ＆トレース環境の構築方法について説明する予定でした。

しかし、前回の記事へのコメントで、「SOEMの動作検証をするためには、結局、EtherCATスレーブを自費で購入しなければならないじゃないか」というのが、ずっと気にかかっていま

した。

　確かに、EtherCATスレーブという、（現時点において）超レアなデバイスを、この連載のためだけに私費で購入するのは難しいだろうなと、私も思います。

　（ところで、EtherCATスレーブを、この連載の読者に無償で貸し出してもよいという太っ腹なベンダー各社の社長さま。ご連絡をお待ちしております。お貸しいただけたら、御社のEtherCATスレーブの製品を、この連載でご紹介させていただく予定です。）

　そこで今回は、当初の予定を変更して、PCベースのメイド（EtherCATスレーブ）の作り方をご説明したいと思います。

　SOEM（EtherCATマスタ）の対向装置となる、PCベースのEtherCATスレーブエミュレータ、名付けて「メイドPC」を提供できれば、スレーブを購入して頂かなくとも、SOEMの動作確認は可能となります。

　まず私は、後輩に、「PCで動くEtherCATスレーブエミュレータを探してほしい」と頼みました。1時間もたたないうちに、彼は、「light ethercat slave」なるものを見つけてきてくれました。

江端：「おお、仕事速いなぁ。ついでに構築もやってくれる？」
後輩：「嫌です」

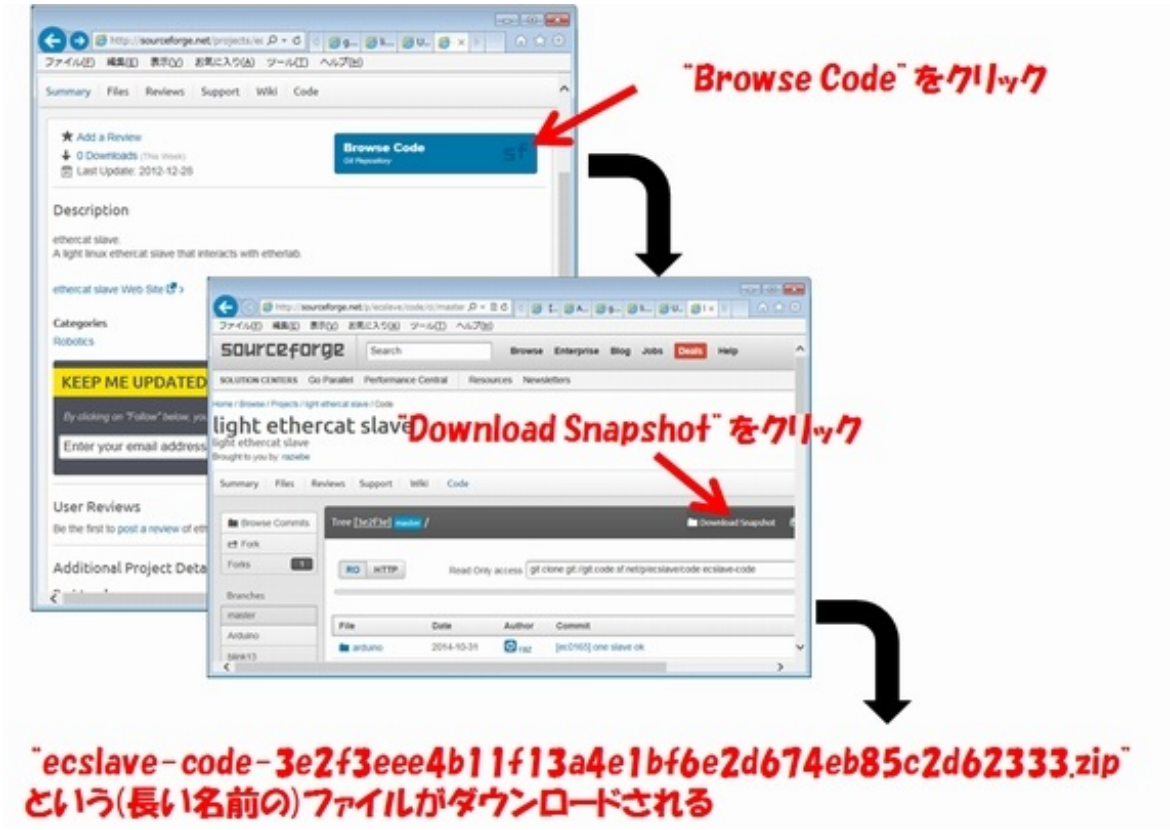
　にべもなく断られましたので、自力で構築を試みました。

　本連載の実験装置は、Windows7の環境で統一したかったのですが、このlight ethercat slaveは、LinuxOSを前提としたオープンソースでした。そこで、今回10年以上前に購入した古いノートPCを探し出して、UbuntuのLinuxの環境で、「メイドPC」を作ることにしました。

　私が構築した手順を記載します。

【Step.1】ノートPCに、「Ubuntu 14.04 LTS 日本語 Remix」をインストールします。私の場合、Windowsの環境はそっくり残して、USBメモリからブートできるようにしました。

【Step.2】light ethercat slaveのソースコードをダウンロードします。



【Step.3】ダウンロードした一式を、適当なディレクトリに持ってきて、(私は、“~/”の直下にしました)解凍します。“ecslave-code-3e2f3eee4b11f13a4e1bf6e2d674eb85c2d62333”というディレクトリ名だと分かりにくいので“ecslave-code”という名前にリネームしておきます。

(以下、こちらの記載に基づいて、淡々と行います。)

【Step.4】構築に必要なツールを、以下の手順でUbuntu Linuxにダウンロードします。

(1) $ sudo apt-get install automake(「Makefile.amからMakefile.inを自動生成するもの」だそうです)

(2) $ sudo apt-get install libpcap-dev(pcapライブラリのインストール)

(3) $ sudo apt-get install libpthread-stubs0-dev(pthreadライブラリのインストール)

【Step.5-1】次に、構築(ビルド)を実行します。

$ cd ecslave-code として、以下を実行(ここから先は、なんでもsudoを付けておけば、安心でしょう)。

(1) $ sudo autoreconf

(2) $ sudo automake --add-missing

（3）$ sudo sh configure

（4）$ sudo make

　ecslave-codeのディレクトリの中に、ex_slaveという名前の実行ファイルができていれば成功です。

　次に、使用方法を説明します。

　書式は以下の通りです。

$ sudo ./ec_slave ethX NUM

（ethXは、イーサネットインタフェースの名前（"eth0", "eth1"など）で、NUMはエミュレートするEhterCATスレーブの数です）

$ ifconfigで、EtherCATマスタ（SOEM）とつないでいるイーサネットインタフェースを確かめた後、

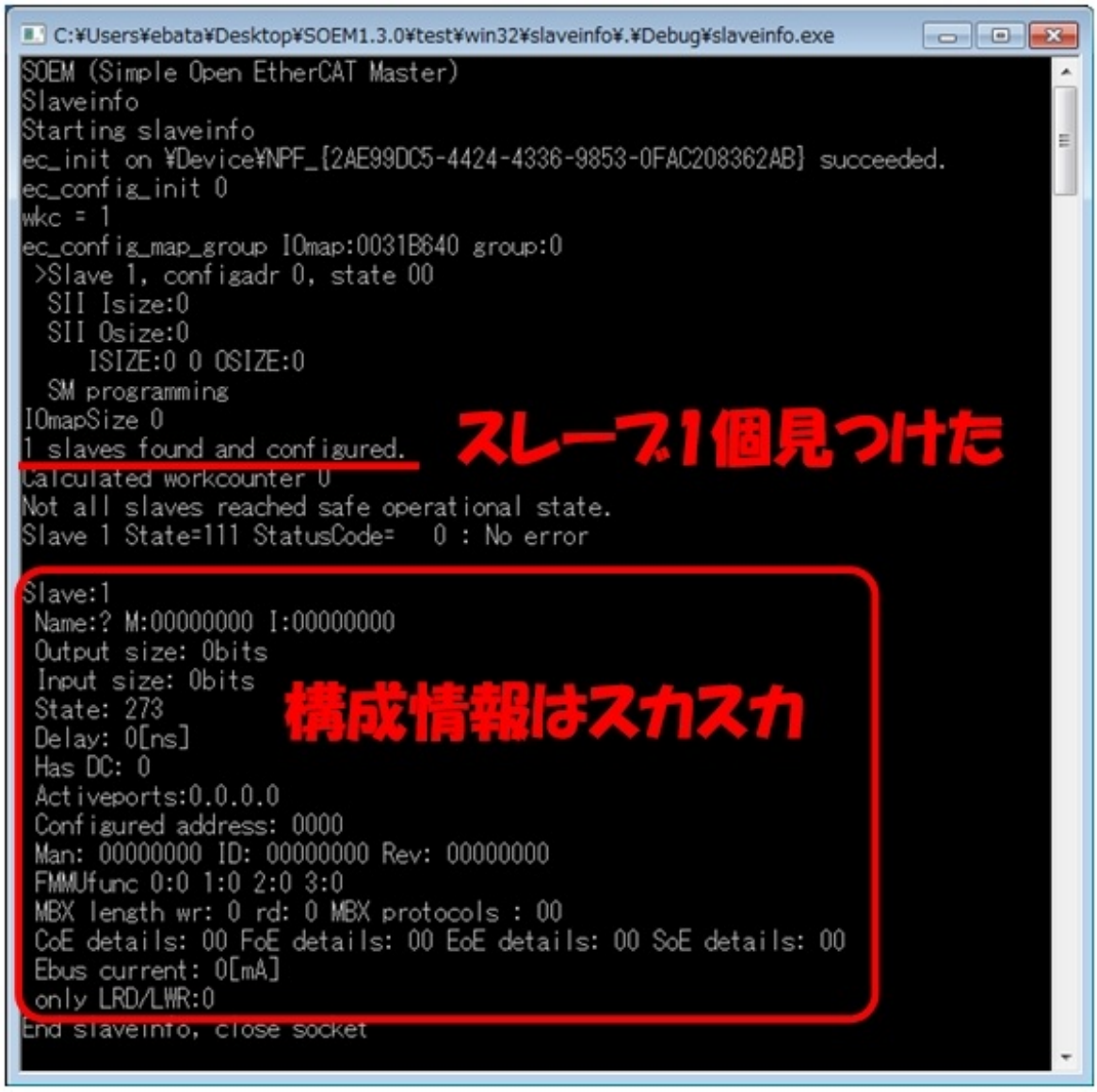
$ sudo ./ec_slave eth0 1

などと入力してください。

eth0が利用可能でない場合は、“LINK 0 eth0 DOWN”
eth0が利用可能の場合は、"LINK 0 eth0 UP"

と表示されます。

　SOEM（Simple Open EtherCAT Master）から、slaveinfo（前回を参照）を実行すると、恐しく時間がかかった後（1分から数分後）に、以下が表示されます。



$ sudo ./ec_slave eth0 3

としてslaveinfoを実行すると、マスタ側に3つ分のスレーブ情報が表示されるので、試してみてください。ただし、表示はさらに遅くなります。これは、「メイドPC」が、Linuxのアプリケーションとして動作しているためです。マスタの動作チェックには使えますが、実際のスレーブの代替にはならないでしょう。

カーネルモードのビルド法

ところで、「light ethercat slave」は、カーネルモードも準備されています。

こちらは、PC丸ごと1台をスレーブとして作り込みますので、応答時間は相当に速くなりますし、スレーブのアプリケーションを実装すれば「メイドPC」を、スレーブとしても使えると思います。

カーネルモードのビルドは以下の通りです。

【Step.5-2】kernelディレクトリに入って、構築（ビルド）を実行します。

$ cd ecslave/kernel

$ sudo make

が、ソースコードに不具合があるため、ビルドに失敗します。

　ですので、以下の6カ所を修正してください。

1. ecslave/kernelにあるMakefileの39行目：
M=$(PWD) → M=$(shell pwd)に変更

2. ecslave/kernelにあるec_slave.cの44行目：
ecat_create_timer() → ecat_create_timer(&ecs)に変更

3. ecslave/kernelにあるecat_timer.cの59行目：
void ecat_create_timer(void) → void ecat_create_timer(ecat_slave * esv)に変更

4. 同じく65行目：
sec = ecat_get_dcstart(0) → sec = ecat_get_dcstart(0, esv)に変更

5. ecslave/stackにあるecat_timer.hの22行目：
void ecat_create_timed(void) → void ecat_create_timed(struct __ecat_slave__ *)に変更

6. ecslave/stack にあるec_sii.cの全部のprintfをec_printfに変更

　再び、$ sudo makeをすると、ecslave/kernelに、ecslave.koができます。

　ここでifconfigを実行して、ハードウェアアドレス（MACアドレス）を読み取ります。

　例えば、00:0b:97:32:c0:c1であるなら、

$ sudo insmod ecslave.ko rxmac=00:0b:97:32:c0:c1

と入力すれば、スレーブとして動作します。

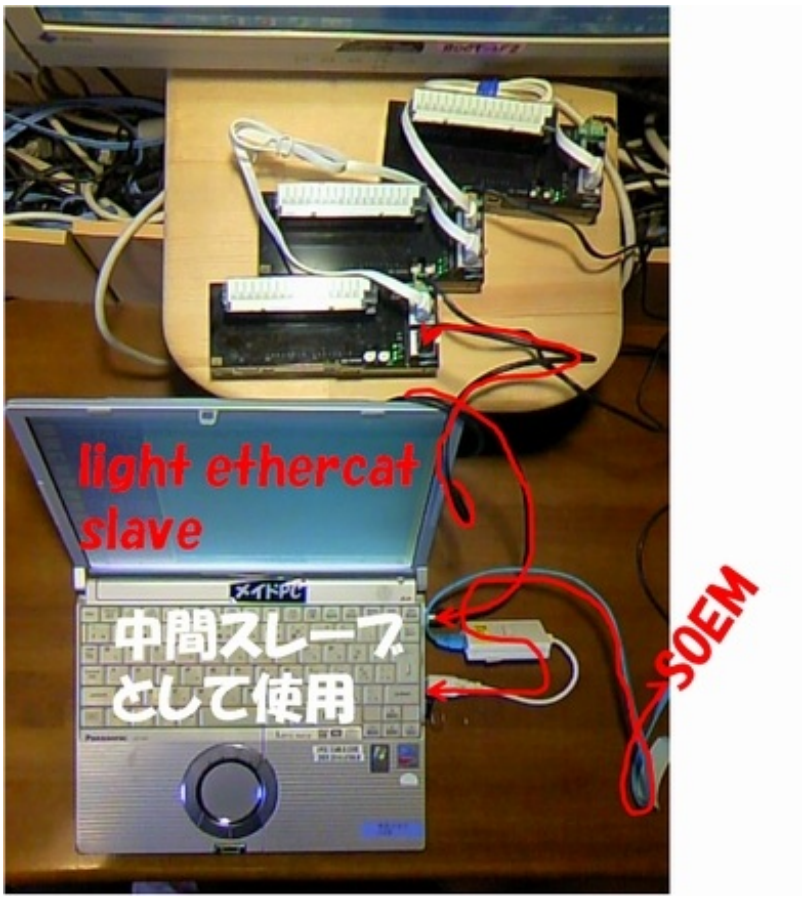
　2つのNICがあれば、もし、デイジーチェーンの終端スレーブとしてだけでなく、中間スレーブとしても使えます。この場合の入力例は、以下の通りです。

$ sudo insmod ecslave.ko rxmac=00:0b:97:32:c0:c1 txmac=00:13:ce:83:d6:ce

　EtherCATスレーブ製品と一緒に動作させてみましたが、動きました（同期外れも発生しましたが）。

ただ、「light ethercat slave」は、EhterCATマスタの性能検証用のエミュレータとして作成されたようで、（DIOなどの）アプリケーションの機能は持っていないようです。

Readmeを読んでみると、「（やりたければ）がんばって実装しろよ」みたいなことが書いてあります（実装された方は、ぜひ私にコードを見せてください）。



今回のコラムを読み直してみると、結局、今回は、EtherCATのメールボックス通信と、遷移状態の話、EtherCATスレーブのエミューレータ（light ethercat slave）の作り方のお話しかできませんでした。

「全自動育児機」や「全自動寝たきり老人介護機」や「老人ホーム4.0」などの話をしなければ、もっとEtherCATの技術的な話を盛り込めるとは思うのです。

しかし、この連載を続けるためには、この連載を読んでいただく皆さんはもちろんですが、それ以上に、この「私」が楽しく書き続けられることがとても重要なのです。そこのところ、何とぞご理解いただけますようお願いします。

それに、「老人ホーム4.0」の原稿を受けとってくれる編集部って、今のところEE Times Japanさんだけですから（と、この連載を打ち切られないように、ヨイショしておきます）。

それでは、来月またお会いしましょう。

⇒本連載のバックナンバーはこちら

特別協力:

本連載では、スレーブの提供などでベッコフオートメーションにご協力いただいております。

Profile

江端智一(えばた ともいち)

日本の大手総合電機メーカーの主任研究員。1991年に入社。「サンマとサバ」を2種類のセンサーだけで判別するという電子レンジの食品自動判別アルゴリズムの発明を皮切りに、エンジン制御からネットワーク監視、無線ネットワーク、屋内GPS、鉄道システムまで幅広い分野の研究開発に携わる。

意外な視点から繰り出される特許発明には定評が高く、特許権に関して強いこだわりを持つ。特に熾烈(しれつ)を極めた海外特許庁との戦いにおいて、審査官を交代させるまで戦い抜いて特許査定を奪取した話は、今なお伝説として「本人」が語り継いでいる。共同研究のために赴任した米国での2年間の生活では、会話の1割の単語だけを拾って残りの9割を推測し、相手の言っている内容を理解しないで会話を強行するという希少な能力を獲得し、凱旋帰国。

私生活においては、辛辣(しんらつ)な切り口で語られるエッセイをWebサイト「こぼれネット」で発表し続け、カルト的なファンから圧倒的な支持を得ている。また週末には、LANを敷設するために自宅の庭に穴を掘り、侵入検知センサーを設置し、24時間体制のホームセキュリティシステムを構築することを趣味としている。このシステムは現在も拡張を続けており、その完成形態は「本人」も知らない。

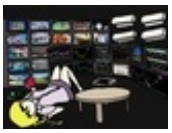
本連載の内容は、個人の意見および見解であり、所属する組織を代表したものではありません。

関連記事

[iOSで作るスマートハウス、AppleのIoT戦略]
次期iOSとなる「iOS 8」には、スマートハウスを構築するための機能「HomeKit」が追加される。スマートハウスにはGoogleも乗り出しているが、Appleは、IoT（モノのインターネット）機器に「MFi」認証を与えることで、Android勢よりもスマートハウス向けネットワークを構築しやすい環境を整えようとしている。

[サムスン電子、安全・音声制御機能を強化したスマートホームサービスをIFAで初公開]
サムスン電子は2014年8月、ドイツ・ベルリンで2014年9月5日（現地時間）に開催される国際コンシューマ・エレクトロニクス展「IFA2014」で、「サムスン スマートホーム」とプレミアム家電の新製品を初公開する。

[人類は、“ダイエットに失敗する”ようにできている]
今回から新シリーズとしてダイエットを取り上げます。ダイエットーー。飽食の時代にあって、それは永遠の課題といっても過言ではないテーマになっています。さて、このダイエットにまつわる「数字」を読み解いていくと、実に面白い傾向と、ある1つの仮説が見えてきます。



[“電力大余剰時代”は来るのか（前編）～人口予測を基に考える～]
今の日本では、「電力が足りる/足りない」は、常に議論の的になっています。しかし、あと十数年もすれば、こんな議論はまったく意味をなさず、それどころか電力が大量に余る時代が到来するかもしれません。

ITmedia Inc.