Politechnika Śląska

Wydział Automatyki, Elektroniki i Informatyki

Kierunek Informatyka

Praca dyplomowa magisterska

Projekt i realizacja stanowiska laboratoryjnego do badania zależności czasowych w sieci EtherCAT

Autor: Damian Karbowiak

Kierujący pracą: dr inż. Jacek Stój

Gliwice, 3 września 2013

Spis treści

1 Wstęp

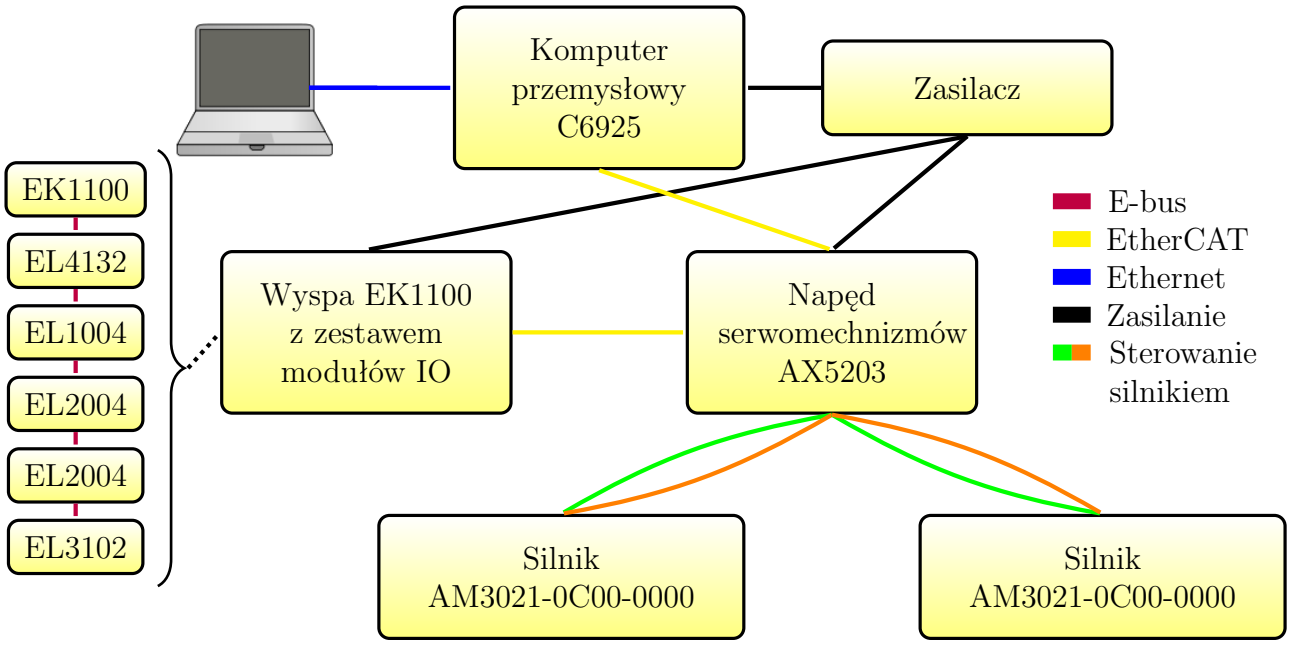
Tematem projektu, którego dotyczy ta praca jest: „Projekt i realizacja stanowiska laboratoryjnego do badania zależności czasowych w sieci EtherCAT". Zagadnienia związane z tworzeniem oprogramowania dla sterowników przemysłowych są dla autora niezwykle interesujące, a zrealizowany projekt miał na celu dalsze pogłębienie jego wiedzy z tego zakresu. Wyboru tego konkretnego tematu autor dokonał, ponieważ protokół EtherCAT jest jeszcze nowością i według wielu źródeł stanowi przyszłość branży informatyki przemysłowej [6, 7], a praca nad tym tematem wydaje się być pomocna i wartościowa w przyszłej pracy zawodowej lub na ewentualnym dalszym etapie kształcenia.

1.1 Stanowisko laboratoryjne

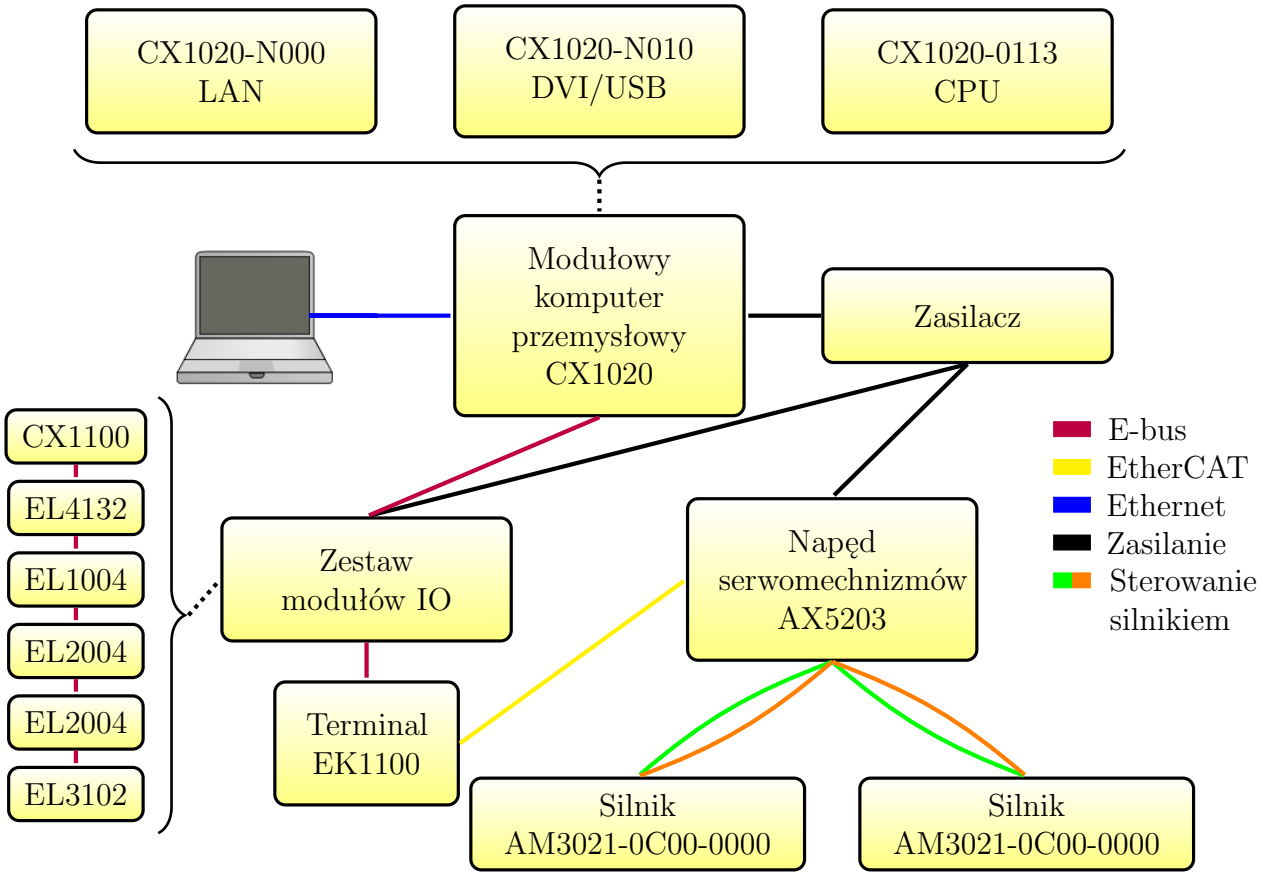
Na potrzeby realizacji projektu wykorzystano dwa różne istniejące stanowiska laboratoryjne, które składały się z elementów opisanych w Tabeli 1.

Stanowisko typu CP (Rysunek 1)	Stanowisko typu CX (Rysunek 2)
1. 2 silniki AM3021-0C00-0000, 2. Wyspa EK1100 z zestawem modułów IO: • Terminal sieci EtherCAT EK1100, • 2-kanałowy moduł wyjść analogowych EL4132, • 4-kanałowy moduł wejść cyfrowych EL1004, • 2 4-kanałowe moduły wyjść cyfrowych EL2004, • 2-kanałowy moduł wejść analogowych EL3102, 3. Napęd serwomechnizmów AX5203 (2 osiowy), 4. Komputer przemysłowy C6925, 5. Zasilacz.	1. 2 silniki AM3021-0C00-0000, 2. Zestaw modułów IO: • 2-kanałowy moduł wyjść analogowych EL4132, • 4-kanałowy moduł wejść cyfrowych EL1004, • 2 4-kanałowe moduły wyjść cyfrowych EL2004, • 2-kanałowy moduł wejść analogowych EL3102, 3. Terminal sieci EtherCAT EK1100, 4. Napęd serwomechnizmów AX5203 (2 osiowy), 5. Modułowy komputer przemysłowy CX1020: • Interfejs USB/DVI CX1020-N010 , • Ethernet CX1020-N000, • CPU CX1020-0113, • Zasilacz CPU i magistrali I/O CX1100, 6. Zasilacz.

Tablica 1: Dostępne stanowiska laboratoryjne



Rysunek 1: Schemat stanowiska typu CP



Rysunek 2: Schemat stanowiska typu CX

1.1.1 Sterownik PLC

W realizacji wykorzystane zostały stanowiska firmy Beckhoff wyposażone w jednostki centralne pracujące pod kontrolą Windowsa CE (Microsoft Windows Compact Edition). Na jednostce takiej uruchamiane są programy do sterowania z poziomu komputera (ang. Soft PLC). Jest to rozwiązanie alternatywne dla klasyczny sterowników swobodnie programowalnych w postaci dedykowanego urządzenia (ang. Hard PLC), nazywanych przez niektórych prawdziwymi (ang. True PLC). Koncepcja ta powstała i jest rozwijana, ponieważ te klasyczne sterowniki posiadają zbyt małe możliwości obliczeniowe oraz szybkość działania jednostki centralnej. W tradycyjnych rozwiązaniach niestety zwiększanie tych możliwości (ilość dostępnej pamięci oraz szybkości działania) powoduje bardzo szybki wzrost ceny gotowego urządzenia. Niezbędnym elementem konfiguracji zestawu, który przekształcamy w „soft PLC" jest karta komunikacyjna umożliwiająca połączenie sterownika z modułami sygnałowymi i wykonawczymi na obiekcie z wykorzystaniem sieci przemysłowej. Tego typu podejście i rozwiązanie ma następujące zalety:

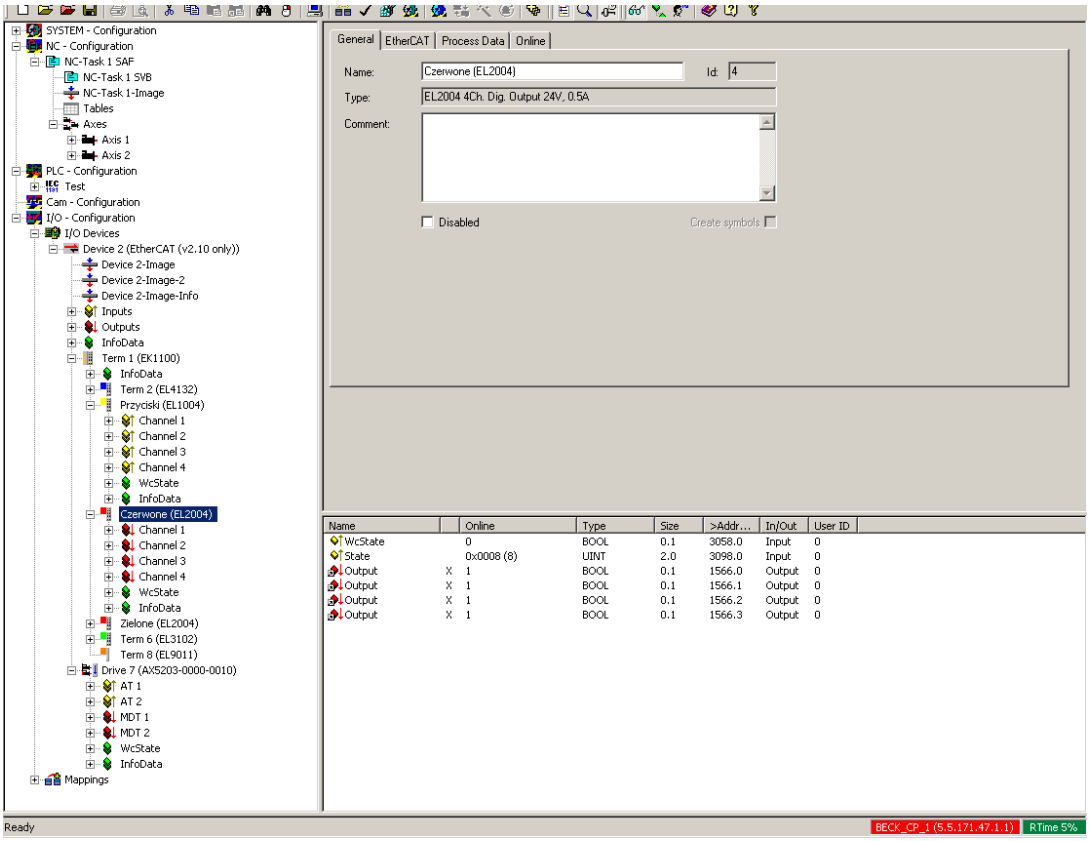
- duże zwiększenie możliwości obliczeniowych przy stosunkowo niewielkim wzroście kosztów,
- możliwość integracji PLC i systemu SCADA na jednym urządzeniu (podobnie jak w panelach operatorskich ze zintegrowanymi sterownikami PLC),
- możliwość zastosowania istniejącej infrastruktury na obiekcie w przypadku przebudowy, należy jedynie podmienić istniejący sterownik typu „hard" na jednostkę wyposażoną w odpowiedni moduł komunikacyjny,
- teoretycznie możliwość zastosowania istniejącego oprogramowania z jednostki „hard PLC", po modyfikacji ewentualnych różnic między systemami.

Taki „sterownik PLC w komputerze PC" wykorzystuje standardowe języki programowania sterowników PLC (zgodność z normą IEC 61131-3) do tworzenia logiki sterującej takiej jak:

- IL – **I**nstruction **L**ist to tekstowy język programowania składający się z serii instrukcji, z których każda zaczyna się z nowej linii i zawiera operator z jednym lub więcej argumentem (zależnie od funkcji),
- LD – **L**adder **D**iagram jest graficznym językiem programowania, który swoją struktura przypomina obwód elektryczny. Doskonały do łączenia POUs (Progam Organization Units). LD składa się z sieci cewek i styków ograniczonej przez linie prądowe. Linia z lewej strony przekazuje wartość logiczną TRUE, z tej strony zaczyna się też wykonywać linia pozioma.
- FBD – **F**unction **B**lock **D**iagram jest graficznym językiem programowania przypominającym sieć, której elementy to struktury reprezentujące funkcje logiczne bądź wyrażenia arytmetyczne, wywołania bloków funkcyjnych itp.
- SFC – **S**equential **F**unction **C**hart to graficzny język programowania, w którym łatwo jest ukazać chronologię wykonywania przez program różnych procesów.

- ST – **S**truktured **T**ext jest tekstowym językiem programowania, złożonym z serii instrukcji takich jak If..then lub For...do.
- CFC – **C**ontinuous **F**unction **C**hart jest graficznym językiem programowania, który w przeciwieństwie do FBD nie działa w sieci, a w luźno poło onej strukturze, co pozwala na np. stworzenie sprzężenia zwrotnego.

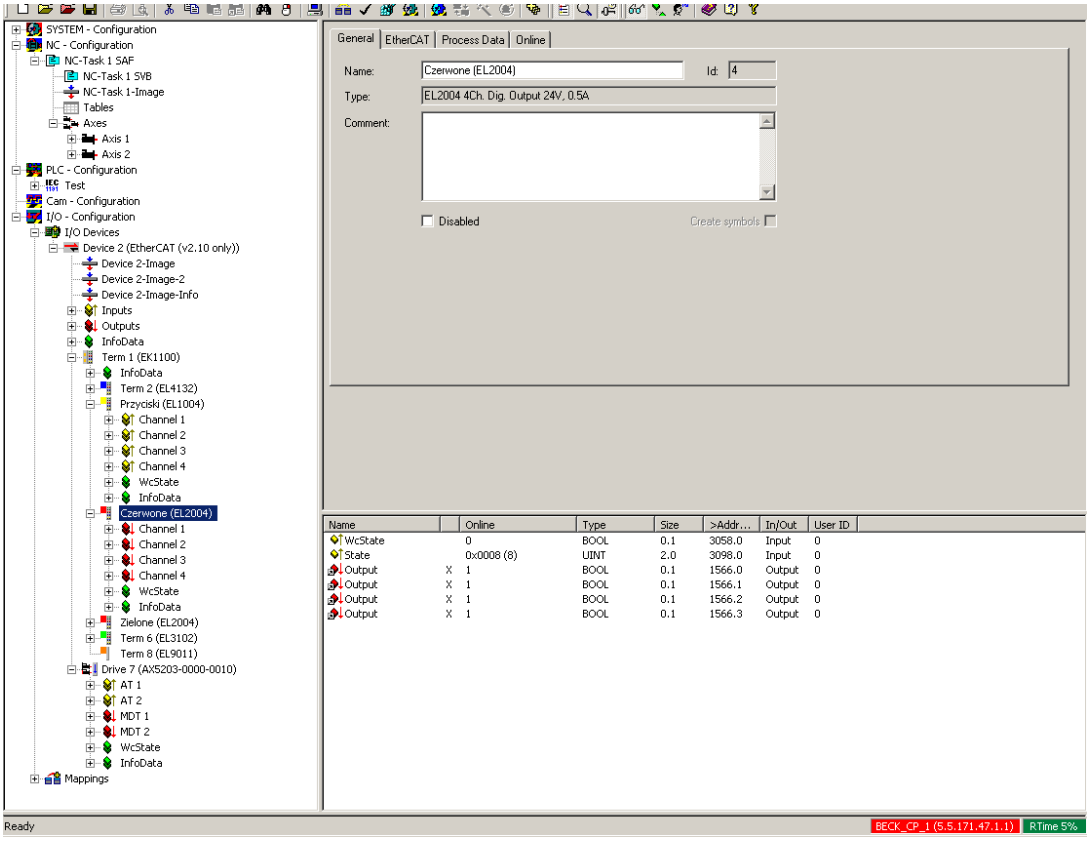
Stanowiska podłączone są do sieci lokalnej Ethernet w laboratorium, więc komunikacja z nimi odbywa się tak samo jak z każdym innym urządzeniem sieciowym. Podstawy programowania i korzystania ze sterowników autor poznał zapoznając się z odpowiednią literaturą [1, 2, 3, 4, 5] oraz uczęszczając w toku studiów na zajęcia obowiązkowe oraz specjalizacyjne. Konfigurację sterowników wraz z modułami przedstawiają Rysunki 3 oraz 4.



Rysunek 3: Konfiguracja stanowiska typu CP

1.1.2 Komputer

Projekt w całości był realizowany na laptopie autora, podłączanym do sieci w laboratorium. Na komputerze uruchomiana była maszyna wirtualna. Na jednej zainstalowane było środowisko TwinCAT do programowania sterownika oraz do tworzenia i uruchamiania wizualizacji. Wizualizacje tworzone w środowisku TwinCAT można uruchomić bezpośrednio na komputerze wyposażonym w odpowiednie oprogramowanie lub na urządzeniu docelowym po podpięciu do niego monitora (o ile urządzenie docelowe posiada wyjście DVI lub odpowiedni interfejs systemowy w postaci odrębnego modułu).



Rysunek 4: Konfiguracja stanowiska typu CX

1.2 Analiza tematu

Analiza tematu polegała przede wszystkim na zapoznaniu się z narzędziami programistycznymi do tworzenia oprogramowania sterownika oraz wizualizacji. W wyniku analizy autor poznał podstawy obsługi środowiska TwinCAT oraz jego elementów składowych a w szczególności:

- TwinCAT System Manager - centralne narzędzie konfiguracyjne,
- TwinCAT PLC - narzędzie do tworzenia programów,
- TwinCAT NC/CNC - grupa narzędzi do sterowania osiami w różnych trybach.

Dodatkowo autor przeczytał wiele artykułów polsko oraz anglojęzycznych poświęconych właśnie protokołowi EtherCAT. Kilka z nich (starszych) traktowało go jako coś bardzo przyszłościowego i obiecującego, natomiast pozostała część opisywała go już jako coś co aktualnie bardzo szybko popularyzuje się i usprawnia procesy przemysłowe.

1.3 EtherCAT

EtherCAT jest nowoczesnym protokołem sieciowym przeznaczonym do stosowania w aplikacjach przemysłowych, szczególnie takich, które wymagają działania całego systemu w czasie rzeczywistym. Nazwa standardu jest skrótem od hasła: „Ethernet for Con-

trol Automation Technology". W zakresie warstwy fizycznej bazuje na Ethernecie. Dodatkowo zaimplementowano w!nim mechanizmy w zakresie organizacji transmisji danych pozwalające na!ominięcie głównych ograniczeń sieci Ethernet. Dzięki temu EtherCAT jest obecnie jednym z popularniejszych oraz szybciej rozwijających się protokołów komunikacyjnych w przemyśle.

1.3.1 Determinizm, przepustowość

1.3.2 Przetwarzanie „w locie"

Zamiast dzielić czas transmisji na dane priorytetowe i mniej ważne lub nadawać priorytety wszystkim pakietom, skorzystano z faktu, że większość danych i rozkazów transmitowanych przez urządzenia przemysłowe wymagające krótkich czasów reakcji jest bardzo mała. W praktyce rozmiar porcji informacji wymienianych jednorazowo przez pojedyncze przyrządy podłączone do sieci jest nawet mniejszy niż minimalny rozmiar ramki stosowanej w klasycznym Ethernecie. Korzystając z tego faktu, twórcy standardu EtherCAT potraktowali ramkę ethernetową jak pewnego rodzaju pamięć RAM, do której zapisywane i z której odczytywane są dowolne informacje w oparciu o adresy lokalizujące pożądane dane w tej pamięci. Kolejne ramki nadawane są przez urządzenie pełniące rolę kontrolera i przesyłane do najbliższego urządzenia podrzędnego. Urządzenie to ma przydzielony adres, który jednoznacznie identyfikuje pewien fragment ramki (zwany datagramem), dzięki czemu może odczytać przeznaczone dla siebie dane. W przypadku gdy urządzenie podrzędne samo chce zainicjować komunikację, zapisuje pod odpowiednim adresem w odebranej ramce informacje przeznaczone dla innego urządzenia podłączonego do sieci EtherCAT. Niezależnie od tego, czy urządzenie coś zapisało, czy też tylko odczytywało fragment ramki, przesyła ją dalej do kolejnego z urządzeń. Duża szybkość tej metody transmisji wynika m.in. z tego, że nie ma potrzeby dekodowania całej ramki danych ani enkapsulacji w protokoły TCP/IP danych zapisywanych. Pozwala to błyskawicznie przekazywać ramki pomiędzy urządzeniami.

(ang. Pooling Timeslicing), (ang. Broadcast Master/Slave)

Wydajność tego rozwiązania jest dosyć duża, choć zależy od liczby elementów podłączonych do sieci. W praktyce czas opóźnienia nie wzrasta powyżej 1 ms i zazwyczaj jest znacznie (kilku- lub kilkunastokrotnie) krótszy. Czas synchronizacji nie przekracza 1 μs.

1.3.3 Transmisja i synchronizacja w sieciach EtherCAT

Jako medium komunikacyjne w sieciach EtherCAT można wykorzystać kable miedziane (100Base-TX), światłowody (100Base-FX) lub łącze E-bus w technologii LVDS. Te ostatnie wprowadzono ze względu na to, że transmisja w sieciach EtherCAT często jest realizowana na krótkich dystansach - E-bus sprawdza się w realizacji łączności na odległość do około 10 m. Kable miedziane sprawdzają się na większych odległościach nieprzekraczających 100 metrów, natomiast użyteczna długość światłowodów może dochodzić aż do 20 kilometrów.

1.3.4 Telegram EtherCAT

Jako pokazano na Rysunku 7, telegram EtherCAT jest inkapsulowany w ramce Ethernet i zawiera jeden lub więcej datagramów EtherCAT dostarczanych do urządzeń slave.

Każdy datagram EtherCAT jest komendą, która zawiera zgodnie z Rysunkiem 8 nagłówek, dane i licznik roboczy. Nagłówek i dane są używane do wyspecyfikowania operacji, które muszą być wykonane przez slave, natomiast licznik roboczy jest aktualizowany przez slave informując mastera, że slave odebrał i przetwarza komendę.

1.3.5 Synchronizacja

1.3.6 EtherCAT Technology Group

Rysunek 5: Logo EtherCAT Technology Group (http://www.ethercat.org).

Standard EtherCAT został opracowany w 2003 roku przez Beckhoff Automation, niemiecką firmę z branży automatyki przemysłowej. Następnie powołano organizację EtherCAT Technology Group (ETG), która zajęła się standaryzacją tego protokołu. Stowarzyszenie to obecnie zajmuje się też organizowaniem szkoleń oraz popularyzacją tego standardu.

Aktualnie w skład ETG wchodzi ponad 2480 firm (dane na dzień 1 września 2013). Najważniejszym członkiem organizacji jest oczywiście firma BECKHOFF Automation. Pozostałe duże i znane firmy wchodzące w jej skład to między innymi: ABB, Brother Industries, BMW Group, Częstochowa University of Technology, Epson, FANUC, Festo, GE Intelligent Platforms, Hitachi, Hochschule Ingolstadt, Mitsubishi, Microchip Technology, Mentor Graphics, Nikon, National Instruments, OLYMPUS, Panasonic, Rzeszów University of Technology, Red Bull Technology, Samsung Electronics, TRW Automotive, Volvo Group, Volkswagen oraz Xilinx.

Jak widać na powyższej liście w skład organizacji wchodzą firmy z bardzo wielu branż, a nawet ośrodki naukowe. Autor pracy wybrał duże i dobrze znane sobie firmy, aby pokazać jak wiele firm interesuje się rozwojem przemysłowych protokołów komunikacyjnych.

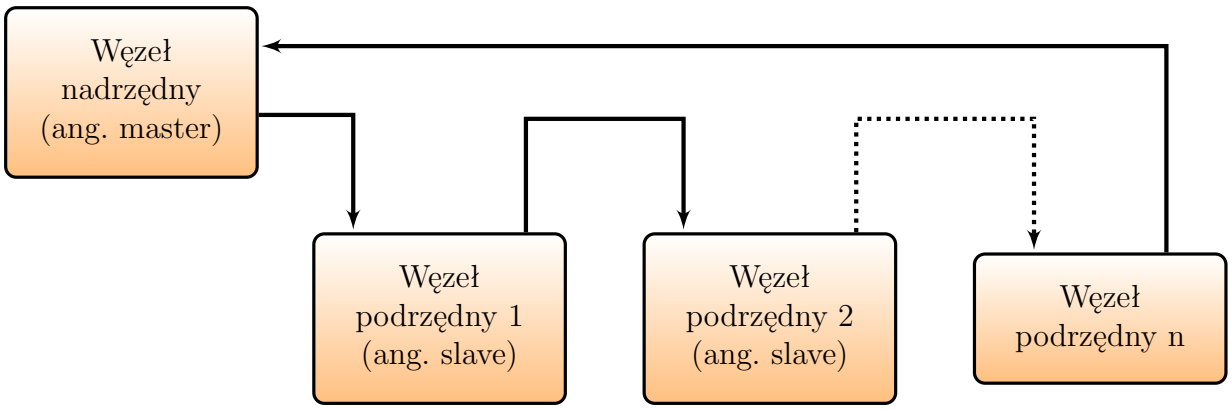
1.3.7

1.3.8

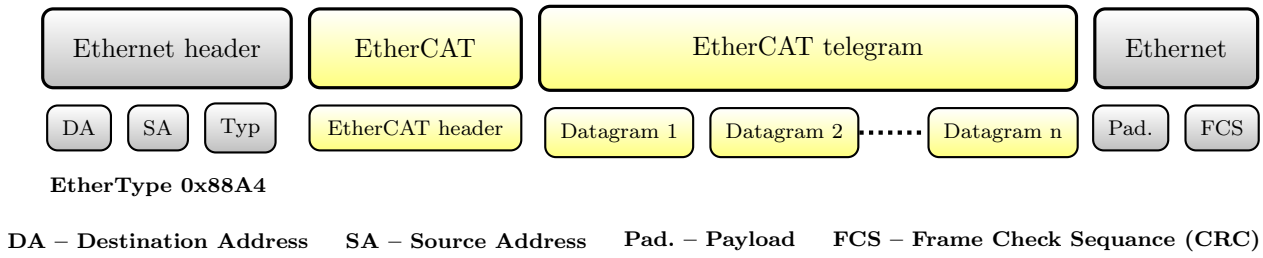
1.4 Plan pracy

Realizacja projektu została podzielona na następujące etapy:

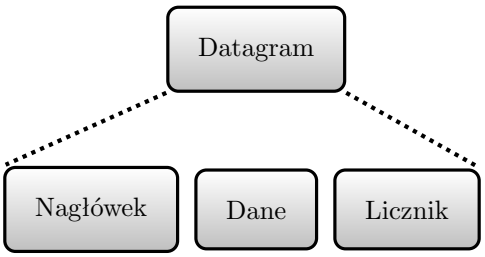
- Zapoznanie się ze sterownikami Beckhoff oraz oprogramowaniem TwinCAT,
- Zapoznanie się z dostępnymi serwonapędami Beckhoff,



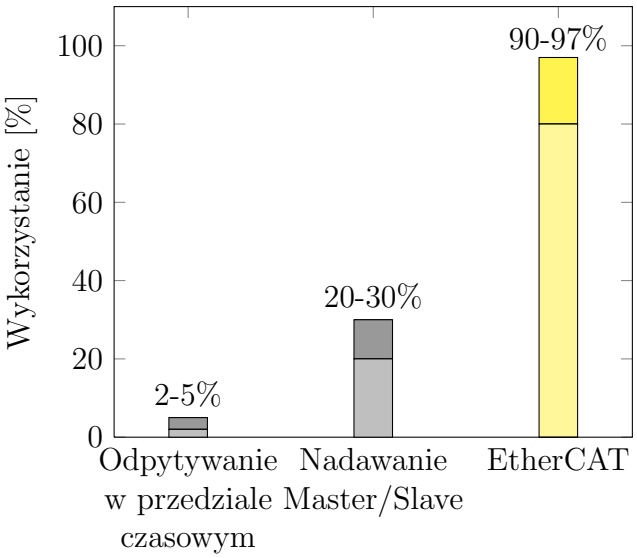
Rysunek 6: Przykładowa topologia sieci



Rysunek 7: Ramka w transmisji EtherCAT i jej podział na datagramy



Rysunek 8: Budowa datagramu



Rysunek 9: Współczynnik wykorzystania kanału transmisyjnego w Ethernecie (dwa pierwsze wykresy od lewej strony) i EtherCAT

- Projekt i realizacja stanowiska,
- Przygotowanie stanowiska do współpracy z systemem wizualizacji,
- Testowanie i uruchamianie,
- Przedstawienie projektu i ewentualne korekty.

Powyższy plan pracy stanowił dla autora wyznacznik kolejnych działań. Jednak powszechnie wiadomo, że w praktyce poszczególne punkty są wymienne i wpływają na siebie wzajemnie.

2 Oprogramowanie sterownika

W niniejszym rozdziale opisane zostało oprogramowanie sterujące modelem. W kolejnych podrozdziałach zostanie przedstawiona specyfikacja zewnętrzna oraz wewnętrzna.

2.1 Specyfikacja zewnętrzna

Specyfikacja zewnętrzna przedstawiona w dalszej części podrozdziału zawiera opis, jak korzystać z oprogramowania wgranego do sterownika przez jego autora. Opisane zostało, jak ustawiać odpowiednie zmienne, aby uzyskać żądany efekt.

Lista zmiennych wejściowych i wyjściowych wymieniana między sterownikiem a modelem została już opisana w pierwszym rozdziale, w Tablicach **??** oraz **??**. Pozostałe zmienne znajdują się w wewnętrznej pamięci sterownika.

Oprogramowanie może sterować modelem w sposób automatyczny lub ręczny. Tryb automatyczny w trybie obsługi magazynu zostanie opisany w podrozdziale 2.1.2. Tryb ręczny może być realizowany przy pomocy zadajnika podpiętego do sterownika lub przy pomocy przycisków umieszczonych na odpowiednim ekranie wizualizacji. W trybie tym o pracy robota decydujący jest stan przycisków. Dopuszczalne są wszystkie możliwe ruchy w przedziale od wyłącznika krańcowego do wartości maksymalnej.

2.2 Specyfikacja wewnętrzna

Podrozdział specyfikacja wewnętrzna opisuje sposób rozwiązania przez autora kwestii sterowania modelem przy użyciu dostępnego na stanowisku sterownika oraz poszczególnych trybów sterowania. W tworzeniu oprogramowania zostały wykorzystane następujące języki programowania:

- język drabinkowy (Ladder), wykorzystany do stworzenia głównych elementów programu,
- S7-SCL, który został zastosowany do korzystania z tablic. Niestety do korzystania z nich nie można zastosować języka LAD, ponieważ nie da się w nim odwoływać do elementów tablicy przez indeksy będące zmiennymi, a jedynie przez stałe. Po zapoznaniu się z dokumentacją okazało się, że taka możliwość istnieje w języku STL, ale jest to metoda skomplikowana w implementacji. Właśnie dlatego najlepszym i najprostszym rozwiązaniem okazuję się S7-SCL, który jest kompilowany do kodu w języku STL.

3 Wizualizacja HMI

Zaimplementowana przez autora wizualizacja ma na celu zobrazowanie działania modelu oraz umożliwienie operatorowi wpływania na jego działanie. Kolejne podrozdziały zawierają opis specyfikacji zewnętrznej oraz wewnętrznej. Część odnosząca się do specyfikacji zewnętrznej jest skróconą instrukcją obsługi użytkownika. Specyfikacja wewnętrzna jest opisem, jak zostały zrealizowane poszczególne elementy i w jaki sposób wizualizacja współpracuje ze sterownikiem.

3.1 Specyfikacja zewnętrzna

Specyfikacja zewnętrzna przedstawiona w dalszej części podrozdziału stanowi skróconą instrukcję obsługi wizualizacji oraz opis możliwości oferowanych przez poszczególne ekrany.

Autor projektu wykorzystał w swojej pracy szereg elementów dostępnych standardowo w środowisku Simatic WinCC flexible. Podstawowymi elementami sterującymi są przyciski w trybie tekstowym oraz przeźroczystym. Głównymi obiektami służącymi do prezentacji informacji są: pola tekstowe, pola wejściowo-wyjściowe oraz pola daty i godziny. Dodatkowo celem uatrakcyjnienia wizualizacji wykorzystane zostały suwaki (ang. *slider*), obrazki oraz zegarek.

Obsługa tej części projektu jest realizowana za pomocą myszy i klawiatury podłączonych do komputera. Za pomocą klawiatury wybieramy interesujący nas ekran lub wprowadzamy żądaną wartość pozycji docelowej na ekranie testowania trybu automatycznego.

3.2 Specyfikacja wewnętrzna

Wizualizacja komunikuje się z komputera klasy PC ze sterownikiem za pośrednictwem protokołu Ethernet w sieci lokalnej. Odniesienia do odpowiednich adresów w pamięci sterownika dokonywane są za pomocą nazw symbolicznych zdefiniowanych w tablicy Tags. Do działania wizualizacja używa tylko jednej zmiennej wewnętrznej i jest to zmienna tablicowa *MagazynDTEnable* z elementami typu bool. Elementy te odpowiadają za wyświetlanie dat oraz godzin na ekranie ze stanem magazynu po kliknięciu na wybraną komórkę. Obsługa wyświetlania dat polega na tym, że po kliknięciu w wybrane pole ustawiana jest odpowiednia zmienna w tej tablicy na wartośc *true*, a po zwolnieniu klawisza myszki na wartość *false*. Za zmiany te odpowiadają niewidzialne przyciski umieszczone na tych polach.

Wizualizacja wpływa na pracę sterownika poprzez zmianę pojedynczych bitów za pomocą umieszczonych na ekranie przycisków. Wpływa ona również poprzez modyfikowanie wybranych zmiennych odpowiadających pozycjom docelowym lub poprzez dodawanie odpowiednich zadań do kolejki. Bardziej zaawansowane operacje zostały zrealizowane za pomocą skryptów napisanych w języku VBScript, które są bardzo prostą i szybką opcją wykonywania bardziej zaawansowanych czynności.

4 Badania

W niniejszym rozdziale opisany został przebieg przeprowadzonych badań oraz analiza ich wyników. W kolejnych podrozdziałach zostaną przedstawione kolejne różne eksperymenty.

4.1 Opóźnienia pojedynczego odcinka sieci

4.2 Wpływ topologi na opóźnienia

4.3 Czas stabilizacji sieci po zmianach

Różne kable Długość kabla

5 Uruchamianie i testowanie

Rozdział ten zawiera podsumowanie przebiegu prac nad projektem. Opisane zostaną tu wszelkie poważne problemy, które wystąpiły w czasie realizacji projektu. Ponadto zawarto tu opis przebiegu procesu testowania.

5.1 Przebieg testowania

W procesie weryfikacji poprawności działania projektu zastosowano testowanie wstępujące.

Głównym testerem był autor projektu więc większość testów przebiegała na zasadzie białej skrzynki (ang. *white box*), bardzo często z użyciem podglądu stanu w środowisku TwinCAT System Manager. Takie testowanie pozwala stosunkowo łatwo wyszukać źródło błędu i je wyeliminować.

Autor kilka razy przeprowadzał testy stosując metodę czarnej skrzynki (ang. *black box*), nie biorąc pod uwagę zależności wykonywanych czynności, od realizowanego przez sterownik kodu. Kilkukrotnie w czasie realizacji projektu do testów zgłaszały się osoby trzecie, które były nim zaciekawione. Testy wykonane przez takie osoby są niezwykle cenne ze względu na dużą nieprzewidywalność oraz całkowitą niezależność działań od rozwiązań ze względu na brak ich znajomości.

W czasie realizacji autor stosował testowanie oparte na dwóch metodach analizy. Testowanie oprogramowania można wykonywać pod kątem analizy statycznej i dynamicznej. Analiza statyczna polega na sprawdzaniu kodu źródłowego i znajdowaniu w nim błędów bez uruchamiania sprawdzanego kodu. Ta metoda była stosowana poza laboratorium, gdzie brak był dostępu do sterownika i modelu. Podczas analizy dynamicznej oprogramowanie jest uruchamiane i badane pod kątem ścieżki przebiegu i czasu wykonywania. Ta metoda z kolei była najważniejsza i często wyniki tych testów były zaskakujące w stosunku do przeprowadzonych wcześniej z zastosowaniem analizy statycznej.

Ostatnim etapem testów były te przeprowadzone w obecności promotora oraz te wykonane przez niego. Ostatecznie oprogramowanie zostało zatwierdzone i uznane za spełniające wszystkie wstępne założenia przedstawione w podrozdziale 1.2.

5.2 Napotkane problemy

Podczas tworzenia projektu napotkane i przeanalizowane zostały następujące problemy:

- Problem z automatycznym uruchamianiem stworzonego projektu PLC:

  Po utworzeniu oprogramowania sterownika PLC oraz po odpowiednim skonfigurowaniu go w oprogramowaniu TwinCAT System Manager tj. linkowaniu zmiennych programu do odpowiednich fizycznych wejść oraz wyjść modelu oraz aktywowaniu tak przygotowanej konfiguracji, które z kolei wymusza zresetowanie systemu nie następuje uruchomienie projektu sterownika PLC. W początkowej fazie autor przełączał się na oprogramowanie TwinCAT PLC Control gdzie logował się do sterownika i ręcznie uruchamiał stworzony przez siebie kod. Niestety to rozwiązanie na dłuższą metę okazało się męczące i czasochłonne. Okazało się, że w sterownikach firmy Beckhoff trzeba w specjalny sposób przygotować oprogramowanie, które ma być uruchamiane w sposób automatyczny, tzn. trzeba utworzyć projekt, który jest bootowalny. Początkowo takie podejście wydało się autorowi bardzo dziwne, ale po dłuższym zastanowieniu oraz kilku rozmowach z bardziej doświadczonymi w branży osobami okazało się, że ma ono swoje plusy. Przykładowo w przypadku tworzenia oprogramowania w fazie rozwojowej reset urządzenia pozwala przerwać całkowicie wykonywanie oprogramowania zawierającego błędy mające destrukcyjny wpływ na model lub w praktyce na obiekt przemysłowy. Po zastosowaniu nowej metody uruchamianie i testowanie tworzonego oprogramowania stało się zdecydowanie prostsze.

- Problem z wykryciem jednego z silników:

  aa

Wszystkie problemy zostały rozwiązane i w ostatecznej wersji oprogramowania nie wpływają one w negatywny sposób na pracę modelu.

6 Wnioski

Protokół EtherCAT jest rozwiązaniem zdecydowanie bardzo nowoczesnym, zaawansowanym oraz pomysłowym. Pozwala bardzo dobrze wykorzystać wzrastające moce obliczeniowe sterowników PLC, przy zachowaniu wszelkich rygorów czasowych. Zdecydowanie ogromny wpływ na tak szybki rozwój ma fakt, że technologia jest bardzo otwarta i postawiona na współpracę wszystkich zainteresowanych rozwojem stron. Olbrzymim plusem jest fakt wykorzystania standardowej struktury Ethernetu co upraszcza proces integracji w obrębie jednego systemu obu standardów.

Temat zdecydowanie nadaje się do pogłębiania i dalszych badań. Autor dopuszcza taką możliwość w ramach prac badawczych w toku swoich studiów doktoranckich. Zarówno same sterowniki firmy Beckhoff oparte o koncepcję „soft PLC" jak i sam badany protokół EtherCAT są na tyle rozbudowane i rozwojowe, że zawsze znajdzie się jakiś aspekt do przeanalizowania od strony teoretycznej i eksperymentalnej. W momencie powstawania niniejszej pracy dwa bardzo interesujące kwestie wydają się autorowi ciekawą postawą do przeprowadzenia badań.

Po pierwsze przetestowanie elementów sieci EtherCAT pochodzących od innych producentów.

IMPLEMENTACJA U INNYCH PRODUCENTÓW PORÓWNANIE - ASIC ARM¡ SINTARA ITD.

Drugim ciekawym rozwiązaniem i pomysłem na które autor natrafił w sieci w czasie analizy tematu, rozwiązywania problemów oraz pisania niniejszej pracy jest EtherLab. Jest to technologia łącząca sprzęt i oprogramowanie w celach testowych oraz do sterowania procesów przemysłowych. Jest to niejako technika zbudowana z dobrze znanych i niezawodnych elementów. EtherLab pracuje jako działający w czasie rzeczywistym moduł jądra otwartego systemu Linux, który komunikuje się z urządzeniami peryferyjnymi poprzez protokół EtherCAT. Rozwiązanie jest całkowicie darmowe i otwarte co na pewno jest jego olbrzymią zaletą. Można zdecydować się na pobranie sobie wszystkich komponentów i ich samodzielne uruchomienie lub zakup gotowego preinstalowanego zestawu startowego bezpośrednio od twórców. Oprogramowanie całego zestawu może zostać wygenerowane przy użyciu Simulinka/RTW lub napisane ręcznie w C. Następnie tak przygotowane jest uruchamiane w środowisku kontrolującym proces (jądro Linuksa oraz moduł czasu rzeczywistego) komunikującym się z „obiektem przemysłowym" poprzez EtherCAT. Dodatkowo można rozszerzyć możliwości całego zestawu poprzez Ethernet TCP/IP dołączając interfejs użytkownika (ang. Frontend) w wersji dla Linuksa lub Windowsa albo jeden z innych dodatkowych serwisów. Przykładowe serwisy to:

- Raportowanie poprzez SMS,
- Zdalne usługi: Internet, ISDN, DSL,
- Usługi sieciowe: Web, DHCP, Drukowanie,
- Logowanie danych (ang. data logging).

Jeżeli autor będzie miał taką możliwość to na pewno chętnie przyjrzy się tej koncepcji ze względu na swoją sympatię do systemu Linux oraz wszystkich rozwiązań go wykorzystujących. Ciekawe wydają się badania wydajności takiego rozwiązania oraz porównanie ich z drogimi rozwiązaniami komercyjnymi.

7 Bibliografia

Literatura, która została wykorzystana przez autora w czasie powstawania projektu, którą opisuje niniejsza dokumentacja.

[1] Jerzy Kasprzyk: *"Programowanie sterowników przemysłowych"*, Wydawnictwa Naukowo-Techniczne WNT, Warszawa, 2007.

[2] *"Programowalne sterowniki PLC w systemach sterowania przemysłowego"*, Politechnika Radomska, Radom, 2001.

[3] Andrzej Maczyński: *„Sterowniki programowalne PLC. Budowa systemu i podstawy programowania"*, Astor, Kraków, 2001.

[4] Zbigniew Seta: *„Wprowadzenie do zagadnień sterowania. Wykorzystanie programowalnych sterowników logicznych PLC."*, MIKOM Wydawnictwo, Warszawa, 2002.

[5] Janusz Kwaśniewski: *„Programowalne sterowniki przemysłowe w systemach sterowania"*, Wyd. AGH, Kraków, 1999.

[6] Andrzej Gawryluk: *„EtherCAT – to nie takie trudne. Ethernet jako sieć real-time"*, Elektronika Praktyczna, 2/2010.

[7] Michał Gosk: *„Szybkość, niezawodność, doskonała synchronizacja. EtherCAT - system przyszłości."*, Magazyn Sensor, 1/2013, kwiecień.

[8] Krzysztof Oprzędkiewicz: *„Realizacja predyktora Smitha na platformie sprzętowo-programowej „soft PLC" bazującej na komputerze klasy PC"*, Automatyka / Akademia Górniczo-Hutnicza im. Stanisława Staszica w Krakowie, 2006, t. 10, z. 2, s. 177–186, ISSN 1429-3447.

8 Spis rysunków, tablic i kodów źródłowych

8.1 Spis rysunków

8.2 Spis tablic

8.3 Spis kodów źródłowych

9 Załączniki

- Oświadczenie o autorstwie,
- Płyta CD, na której znajdują się:
  - Kod oprogramowania wewnętrznego TwinCAT PLC Control,
  - Pliki projektu TwinCAT System Manager,
  - Kod wizualizacji oraz pliki projektu !?,
  - Plik wykonywalny wizualizacji !?
  - LaTeX-owe pliki źródłowe pracy magisterskiej.