



mgr inż. Marcin Michalski,
dr inż. Józef Wiora, e-mail: jozef@wiora.pl
Politechnika Śląska

s. 184–201

DOI: 10.17274/AEZ.2016.26.10

Możliwość zastosowania technologii Internetu rzeczy w przetwornikach pehametrycznych

Application Possibilities of the Internet of Things Technology in pH Transducers

Abstract: The term “Internet of things,” despite the lack of commonly acceptable definition, is now often used, mainly for marketing purposes, by manufacturers of various types of equipment. This article shows how to understand this term based on the recommendations of the International Telecommunication Union (ITU), and gives properties of the discussed technology. Implementation of the technology to a pH measurement transmitter is also described and results of laboratory tests are presented.

Streszczenie: Termin „internet rzeczy”, pomimo braku powszechnie akceptowanej definicji, jest obecnie często wykorzystywany, głównie w celach marketingowych, przez producentów różnego rodzaju urządzeń. W artykule przedstawiono, jak należy rozumieć ten termin na podstawie rekomendacji Międzynarodowego Związku Telekomunikacyjnego ITU, oraz podano cechy omawianej technologii. Opisano również sposób zastosowania internetu rzeczy w przetworniku pehametrycznym, a także zaprezentowano wyniki testów laboratoryjnych.

Keywords: ion-meter, pH, Internet, SCADA

Słowa kluczowe: jonometr, pH, Internet, SCADA

1. WSTĘP

Żyjemy w czasach charakteryzujących się niezwykle dynamicznym rozwojem nauki i techniki. W ścisłym związku z tym postępowaniem pozostają ludzie i ich nowatorska działalność, która staje się najważniejszym wyznacznikiem gospodarczego postępu. Człowiek, chcąc zapewnić sobie dalsze istnienie, panowanie nad przyrodą, ale też nad materialnymi i duchowymi dobrami, które sam wytworzył, dąży do posiadania umiejętności właściwego rozpoznawania i oceniania sytuacji dnia dzisiejszego oraz przewidywania bliższej i dalszej przyszłości.

Obecnie jednym z zagadnień, nad którym trwają intensywne prace na świecie, jest internet rzeczy, który często określa się mianem rewolucji internetowej [1]. Mogłoby się wydawać, że jest on czymś nowym. Jednak według Cisco Internet Business Solutions Group można o nim mówić już od momentu, kiedy liczba rzeczy i obiektów podłączonych do Internetu przekroczyła liczbę ludności na świecie (6 mld). Zdarzenie to datuje się na przełom 2008 i 2009 roku. Internet rzeczy postrzegany jest jako potencjał, który może zmienić świat w większym stopniu niż Internet [2]. Wynika to z możliwości łączenia się nie tylko ludzi, ale również rzeczy (urządzeń) w sieci. Wśród wielu budzi to pewne kontrowersje, ponieważ z jednej strony jest szansą na lepsze jutro, a z drugiej rodzi obawy przed postępującą i coraz bardziej obecną w naszym życiu cyfryzacją [3].

Pomimo wielu prowadzonych w tym zakresie badań wciąż trudno o jednolitą definicję internetu rzeczy. Również standardy budowy urządzeń, rozwiązań programistycznych oraz sposobów zabezpieczenia całej infrastruktury i gromadzonych w niej danych wciąż się krystalizują. W związku z tym w niniejszej pracy przeanalizowano literaturę oraz zreferowano dzisiejszy stan wiedzy i badań dotyczących internetu rzeczy. Szczególnym urządzeniem, w odniesieniu do którego przeprowadzono badania, jest **pehametr**.

2. ELEKTRODY JONOSELEKTYWNE I PEHAMETRIA

Stężenie jonów wodorowych jest znaczącym parametrem zarówno w przyrodzie, jak i przy prowadzeniu procesów przemysłowych [4]. Pojęcie **pH** zostało wprowadzone w 1909 roku przez duńskiego biochemika i fizykochemika Sorena Sorensena, który określił je jako **potentia hydrogenii** (łac.), co oznacza **aktywność jonów wodorowych** [5]. Skalę pomiarową ustalił, używając roztworów o ściśle określonym składzie chemicznym, które nazywa się roztworami

buforowymi (są one wzorcami materiałów) poprzez przypisanie im odpowiedniej wartości **pH**. Niewielkie rozcieńczenie wodą nie powoduje zmiany **pH** bufora, ponieważ jest on chemicznie stabilny [4]. W uproszczeniu wartość **pH** definiuje się jako ujemny logarytm ze stężenia jonów wodorowych [6, 7]:

$$\text{pH} = -\log c_{\text{H}^+}, \quad (1)$$

gdzie c_{H^+} oznacza stężenie jonów wodorowych w mol L^{-1} .

Wartość **pH** można oznaczać metodą potencjometryczną. Stosuje się wtedy elektrody jonoselektywne (ang. *ISE – ion-selective electrode*), których potencjał względem elektrody odniesienia zmienia się pod wpływem zmian aktywności jonów w badanym roztworze [8: Elektroda jonoselektywna]. Należą one do najstarszej grupy czujników chemicznych. Popularnym i najczęściej wykorzystywanym rodzajem elektrody jonoselektywnej jest elektroda szklana, która została opracowana w 1909 roku przez polskiego naukowca Zygmunta A. Klemensiewicza pod kierunkiem niemieckiego noblisty pochodzenia żydowskiego Fritza Habera [8: Fritz Haber, 9]. Elektroda ta do dziś stanowi wygodny i pewny przyrząd do pomiaru stężenia jonów wodorowych [9, 10].

Wartość potencjału elektrody w zależności od **pH** badanego roztworu określa równanie opracowane przez urodzonego w Wąbrzeźnie niemieckiego chemika i fizyka Waltera H. Nernsta, który w 1920 roku otrzymał Nagrodę Nobla w dziedzinie chemii [6, 7, 8: Walther Nernst]:

$$E = E^0 + \frac{RT}{nF} \ln c_{\text{H}^+} = E^0 + 2,3 \frac{RT}{nF} \lg c_{\text{H}^+} = E^0 + 2,3 \text{SpH} , \quad (2)$$

gdzie:

E – potencjał elektrody pomiarowej,

E^0 – potencjał normalny elektrody,

$R = 8,31 \text{ J (mol K)}^{-1}$ – stała gazowa,

T – temperatura bezwzględna,

n – wartościowość jonów,

$F = 9,65 \cdot 10^4 \text{ C mol}^{-1}$ – stała Faraday'a,

S – czułość elektrody [8: Stała gazowa, 8: Stała Faradaya].

Do wykonania pomiaru potrzebne jest ogniwo pomiarowe składające się z dwóch elektrod: pomiarowej (wskaźnikowej), której potencjał zależy od stężenia odpowiednich jonów w badanym roztworze, oraz porównawczej (referencyjnej, odniesienia), która z kolei zapewnia stały potencjał odniesienia. Jako elektrody porównawcze najczęściej stosuje się elektrody chlorosrebrowe, zaś funkcję elektrody pomiarowej pełni elektroda z membraną szklaną [4, 6, 7]. W zastosowaniach przemysłowych często używa się elektrod zespolonych (kombinowanych), które łączą w zwartej obudowie obie elektrody tworzące ogniwo pomiarowe.

Materiał membrany elektrody pomiarowej jest dobierany tak, by była ona selektywna na określony jon. Ze względu na niedoskonałości budowanych elektrod wykazują one także czułość na inne jony. Jest to tak zwana czułość poprzeczna. Wykonanie pomiaru stężenia określonego jonu jest możliwe tylko, gdy stężenia jonów zakłócających w roztworze są stałe lub gdy jony te są z roztworu usuwane. Zachowanie ogniwa pomiarowego w obecności jonów zakłócających opisuje równanie Nikolskiego-Eisenmana [4].

Obecność matrycy zakłóceń na stałym poziomie w roztworze objawia się nieliniowością charakterystyki $E = f(\text{pH})$. Zakrzywienie wspomnianej charakterystyki powoduje również wnikanie sodu w materiał membrany przy pracy elektrody w roztworach zasadowych. Zjawisko to nazywa się efektem sodowym [4].

W praktyce pomiar **pH** jest realizowany za pomocą pehametru. Jest to wysokoimpedancyjny miliwoltomierz wyskalowany w jednostkach **pH**, który powinien cechować się dokładnością pomiaru na poziomie 1 mV przy zakresie $\pm 1400 \text{ mV}$, rezystancją wejściową powyżej $10^{11} \Omega$, regulowaną czułością, aby umożliwić kompensację zmian czułości elektrod pomiarowych, zmianą punktu zerowego, kompensacją wpływu temperatury na ogniwo pomiarowe oraz odpornością na zakłócenia zewnętrzne [7].

Błędy pomiarowe w pehametrze wywoływane są przez sygnały zakłócające i zmiany parametrów wzmacniacza pomiarowego. Duży wpływ mają zakłócenia powstające na rezystancjach elektrod i rezystancji wejściowej

wzmacniacza. Nie bez znaczenia pozostaje pole elektryczne pochodzące od ładunków statycznych, które jest istotnym źródłem zakłóceń. Układ pomiarowy chroni się, ekranując wewnętrznie elektrody, przewody doprowadzające, stanowisko pomiarowe i wyrównując potencjały stanowiska. Zakłócenia mogą być też powodowane zmianą napiężeń mechanicznych w przewodach łączących elektrody z pehametrem (wyginanie). Jeszcze inną wielkością zakłócającą jest pojawianie się prądów izolacji. Chcąc zminimalizować ich wpływ na dokładność pomiaru, należy utrzymać dobrą rezystancję izolacji całego układu, zwracając szczególną uwagę na punkty o wysokim potencjale względem zacisku wejściowego elektrody porównawczej [7].

Charakterystyki elektrod zmieniają się w czasie, a nawet rodziny charakterystyk par elektrod tego samego typu są przesunięte względem siebie. W związku z tym konieczne jest wykonanie wzorcowania każdego zestawu z wykorzystaniem roztworów buforowych [7].

3. INTERNET RZECZY

Pojęcie internet rzeczy (ang. *Internet of Things – IoT*) zostało po raz pierwszy użyte w 1999 roku przez Kevina Ashtona jako tytuł prezentacji, którą prowadził w firmie Procter & Gamble. Zwrócił on uwagę na to, że większość danych obecnych w Internecie jest wprowadzana lub generowana przez człowieka. Zasugerował również, że ze względu na ograniczony czas i dokładność ludzi lepszym rozwiązaniem byłoby zastosowanie czujników podłączonych do sieci, co pozwoliłoby komputerom na samodzielne zbieranie informacji o rzeczywistym świecie [11].

Znaczenie IoT, jak również standardy dotyczące budowy urządzeń według tej koncepcji wciąż się krystalizują. Poniżej przedstawia się przegląd literatury dotyczącej tego pojęcia. Można w nim wyróżnić dwie części: pierwsza zawiera opracowanie na podstawie rekomendacji International Telecommunication Union (ITU – Międzynarodowy Związek Telekomunikacyjny) [8: Międzynarodowy Związek Telekomunikacyjny, 12, 13], natomiast druga stanowi przegląd wybranych publikacji opisujących dotychczas wykonane w tym zakresie badania.

3.1. Rekomendacje ITU

Definiuje się następujące terminy [12, 13]:

- **Aplikacja** – uporządkowany zbiór zdolności, który zapewnia zespół funkcji i usług stanowiący wartość dodaną. Są one wspierane przez jeden lub więcej serwisów, które z kolei mogą być wspierane przez funkcje publiczne aplikacji.
- **Klient** – osoba lub firma. Kupuje usługi i produkty lub otrzymuje je bez opłat. Jeden konsument może oznaczać wielu użytkowników.
- **Serwis/usługa** – zbiór funkcji i udogodnień, które dostawca oferuje użytkownikowi.
- **Dziedzina aplikacji** – obszar wiedzy lub działalności mający zastosowanie do jednego specyficznego pola działania: ekonomia, komercja, społeczeństwo, administracja.
- **Rzecz to obiekt, który daje się zidentyfikować i można go zintegrować z siecią komunikacyjną.** Można rozróżnić rzecz fizyczną, gdy dana rzecz jest obiektem świata fizycznego (środowisko, roboty przemysłowe, sprzęt elektryczny) oraz rzecz wirtualną, w przypadku obiektów świata informacji (oprogramowanie aplikacji, treści multimedialne). Rzeczy fizycznej może odpowiadać jedna lub więcej rzeczy wirtualnych.

Urządzenie w IoT.

Istotnym elementem IoT jest również urządzenie definiowane jako sprzęt, który obowiązkowo musi mieć możliwość komunikowania się oraz może (nie musi) wykonywać pomiary, oddziaływać (np. urządzenie wykonawcze), zbierać dane, przechowywać je i przetwarzać. Dane zebrane przez urządzenia są przez nie przesyłane do sieci w celu dalszego ich przetworzenia. Część urządzeń podejmuje działania na podstawie otrzymanych z sieci informacji.

Urządzenia można podzielić na:

- **przenoszące dane** – urządzenie dołączone do fizycznej rzeczy łączy ją pośrednio z siecią komunikacyjną,
- **zbierające dane** – odczytuje lub zapisuje dane i może oddziaływać z fizyczną rzeczą (pośrednio przez urządzenie przenoszące dane lub bezpośrednio),

- **mierzące i wykonawcze** – zamieniają sygnały z otoczenia na postać cyfrową poprzez detekcję bądź pomiar lub – w przypadku urządzeń wykonawczych – podejmują działania na podstawie sygnałów cyfrowych otrzymanych z sieci,
- **urządzenie ogólne**, które ma moc obliczeniową i potrafi się komunikować.

Warstwa urządzeń zawiera zdolności związane z urządzeniami i bramami sieciowymi. Należą do nich:

- **wymiana danych z siecią komunikacyjną**, która może zachodzić bezpośrednio lub pośrednio z wykorzystaniem bramy sieciowej,
- **tworzenie sieci ad-hoc** w przypadkach, w których zwiększona skalowalność i szybkie rozlokowanie urządzeń są pożądane,
- **usypianie i wybudzanie** przydatne w aplikacjach, w których trzeba oszczędzać energię (np. przy zasilaniu baterijnym).

Specyficznym typem urządzenia jest brama sieciowa. Powinna ona wspierać wiele protokołów, by współpracować z urządzeniami komunikującymi się z wykorzystaniem różnych technologii (przewodowych i bezprzewodowych). Od strony urządzeń mogą to być takie standardy, jak ZigBee, WiFi czy CAN, natomiast po stronie sieci może to być na przykład Ethernet, sieć 2G lub 3G, jak również LTE. Brama sieciowa dokonuje (jeśli jest taka potrzeba) przetwarzania protokołów. To przetwarzanie może zachodzić w warstwie urządzeń, gdy występują w niej różne technologie i związane z nimi protokoły, lub między warstwą urządzeń a siecią komunikacyjną.

Definicja i wymagania dla IoT.

Internet rzeczy to globalna infrastruktura dla społeczeństwa informacyjnego, która umożliwia realizację zaawansowanych usług przez łączenie rzeczy (wirtualnych i fizycznych). Wykorzystuje w tym celu istniejące oraz dopiero ewoluujące interoperacyjne technologie informacyjne i komunikacyjne. IoT w pełni użytkuje rzeczy. Wykorzystuje identyfikację, zbieranie danych, przetwarzanie i komunikację. Dzięki temu oferuje usługi do wszystkich rodzajów aplikacji przy zapewnieniu bezpieczeństwa i prywatności [12].

Przewiduje się, że internet rzeczy zintegruje zaawansowane techniki pomiarów i oddziaływania (urządzeń wykonawczych) z przodującymi technologiami, jak zaawansowana komunikacja między maszynami, autonomiczne sieci, eksploracja danych, podejmowanie decyzji, ochrona prywatności i bezpieczeństwa oraz obliczenia w chmurze. Ma dodać do komunikacji trzecią cechę – prócz „gdziekolwiek” i „kiedykolwiek” dochodzi „cokolwiek”. Komunikacja może się więc odbywać między: komputerami (ang. *PC to PC*), ludźmi (ang. *human to human*), człowiekiem a maszyną (ang. *human to machine*) oraz między maszynami (ang. *machine to machine, thing to thing*). Dziedziny, w których może znaleźć zastosowanie, obejmują: inteligentne systemy transportu, inteligentne sieci elektryczne, e-zdrowie oraz inteligentne domy. Aplikacje mogą być oparte zarówno na własnościowych platformach aplikacyjnych, jak i na powszechnych platformach zapewniających ogólne możliwości (uwierzytelnianie, zarządzanie urządzeniami, pobieranie opłat, konta). Sieci komunikacyjne powinny zapewniać niezawodny i wydajny transfer danych między urządzeniem a aplikacją (lub innym urządzeniem) oraz odwrotnie – z aplikacji do urządzenia. Sieci internetu rzeczy mogą być realizowane na podstawie protokołu TCP/IP lub z wykorzystaniem sieci ewoluujących. Można wyróżnić trzy przypadki komunikacji w IoT:

1. **przez sieć komunikacyjną przez bramę,**
2. **przez sieć komunikacyjną bez bramy,**
3. **bezpośrednio, bez użycia sieci komunikacyjnej.**

Możliwe jest też połączenie przypadków 1 i 3 oraz 2 i 3.

Wśród podstawowych cech internetu rzeczy wyróżnia się:

- **łączność**, ponieważ według tej koncepcji wszystko może być podłączone do ogólnoswiatowej infrastruktury komunikacyjnej i informatycznej,
- **usługi związane z rzeczami**, co jest już sugerowane przez samą nazwę „internet rzeczy”, czyli Internet, w którym więcej komunikują się rzeczy niż ludzie,

- **różnorodność** wynikającą z obecności rozmaitych platform sprzętowych i sieci (komunikacja może zachodzić przez różne rodzaje sieci komunikacyjnych; różne urządzenia i platformy usługowe mogą oddziaływać ze sobą),
- **dynamiczne zmiany**, którym może podlegać zarówno stan jak i liczba urządzeń w sieci oraz ich fizyczna lokalizacja,
- **ogromną skalę**, ponieważ powstaje konieczność zarządzania niesamowitą ilością urządzeń. Przewiduje się, że liczba ta będzie co najmniej o rząd wielkości większa niż w obecnym Internecie. Więcej ruchu w sieci będą generowały urządzenia (rzeczy) niż ludzie.

IoT można też scharakteryzować za pomocą wymagań wysokopoziomowych, takich jak:

- **łączność oparta na identyfikacji** – połączenie między rzeczą a IoT jest zestawiane z wykorzystaniem identyfikatora danej rzeczy. Należy się liczyć z koniecznością przetwarzania różnych identyfikatorów w zuniifikowany sposób,
- **interoperacyjność, która jest potrzebna do dostarczania i użytkowania danych oraz usług w różnorodnych i rozproszonych systemach**,
- **autonomiczne sieci, o których najkrócej można powiedzieć, że dbają o siebie same**. Czynności, które takie sieci wykonują samodzielnie, to: zarządzanie, konfiguracja, leczenie (naprawa), optymalizacja, ochrona. Wszystko to musi być wspierane w funkcjach kontrolujących sieć internetu rzeczy,
- **autonomiczne świadczenie usług, co oznacza, że usługi są świadczone przez automatyczne zbieranie i przetwarzanie danych pochodzących od rzeczy oraz automatyczną komunikację**. Całość działań odbywa się według zasad ustawionych przez administratorów lub skonfigurowanych przez użytkowników. Znaczenie może tu mieć automatyczne łączenie i eksploracja danych,
- **zdolności związane z lokalizacją, które są potrzebne w przypadku usług i komunikacji zależnych od położenia (lokalizacji) rzeczy i/lub użytkownika**. Urządzenie musi automatycznie śledzić zmiany lokalizacji, co wymaga wbudowania odpowiedniego sprzętu. Dodatkowo trzeba zapewnić zgodność z lokalnymi przepisami prawa oraz ochronę prywatności użytkowników,
- **bezpieczeństwo** – poufność, integralność i autentyczność danych oraz usług muszą być zagwarantowane pomimo zagrożeń jakie niesie ze sobą podłączenie wszystkiego do sieci. Poważne wyzwanie stanowi zintegrowanie różnych polityk bezpieczeństwa i technik z nimi związanych,
- **ochrona prywatności** – prywatność należy chronić, przy czym możliwość zidentyfikowania źródła danych musi zostać zachowana,
- **wysokiej jakości i bezpieczeństwa usługi związane z ciałem człowieka** – należy się w tej kwestii stosować do lokalnych przepisów prawa oraz wziąć pod uwagę względy moralne,
- **plug & play** – urządzenie powinno działać poprawnie po podłączeniu bez dodatkowej ingerencji użytkownika w jego konfigurację. Internet rzeczy musi więc wspierać generowanie, kompozycję lub zbieranie semantycznej konfiguracji, aby zapewnić bezproblemową integrację połączonych urządzeń z aplikacjami, jak również odpowiedzieć na potrzeby tych aplikacji,
- **zarządzanie przez odpowiednie osoby** powinno być dostępne, pomimo że aplikacje IoT z reguły działają automatycznie.

Modele internetu rzeczy. Internet rzeczy można zamodelować na dwa sposoby. Pierwszy polega na użyciu modelu referencyjnego, a drugi na zastosowaniu podejścia zgodnego ze zuniifikowanym językiem modelowania (ang. *Unified Modelling Language* – UML) [8].

W modelu referencyjnym IoT wyróżnia się cztery warstwy. Od najwyższego poziomu są to:

- warstwa aplikacji,
- warstwa wsparcia usług i aplikacji,
- warstwa sieci,
- warstwa urządzeń.

Dodatkowo przez wszystkie wyżej wymienione poziomy przechodzą warstwy zarządzania oraz bezpieczeństwa.

W warstwie aplikacji znajdują się aplikacje internetu rzeczy. Warstwa wsparcia aplikacji i usług zawiera dwa rodzaje zdolności: ogólne, z których korzystają różne aplikacje IoT, oraz szczególne, które z kolei są dostosowane do wymagań konkretnych aplikacji. Możliwości szczególne mogą korzystać z ogólnych. Warstwa sieci również zapewnia dwa typy zdolności: sieciowe, czyli funkcje kontrolujące sieć (np. kontrola dostępu i zasobów transportowych, zarządzanie mobilnością, uwierzytelnianie), oraz transportowe, które zapewniają łączność na potrzeby usług i aplikacji IoT, jak również przesyłanie danych związanych z zarządzaniem i kontrolą. Możliwości związane z zarządzaniem obejmują: awarie, konfigurację, zarządzanie kontami, wydajność, bezpieczeństwo.

Wśród podstawowych i ogólnych możliwości zarządzania znajdują się:

- **zarządzanie urządzeniem, w tym zdalne włączenie i wyłączenie, diagnostykę, aktualizację, zarządzanie statusem pracy,**
- **zarządzanie topologią sieci lokalnej,**
- **zarządzanie ruchem i obciążeniem sieci, co obejmuje również wykrywanie stanów przeciążenia i rezerwację zasobów dla komunikacji krytycznej ze względu na czas lub życie.**

Z kolei szczególne możliwości zarządzania zależą od wymagań konkretnej aplikacji.

Funkcje i usługi związane z bezpieczeństwem dzielą się na ogólne i szczególne. Szczególne są związane z konkretną aplikacją, natomiast ogólne zależą od warstwy w modelu referencyjnym IoT:

- **warstwa aplikacji:** autoryzacja, uwierzytelnianie, poufność danych i ich integralność, prywatność, audyt bezpieczeństwa, ochrona antywirusowa,
- **warstwa sieci:** autoryzacja, uwierzytelnianie, poufność, integralność sygnałów,
- **warstwa urządzeń: autoryzacja,** uwierzytelnianie, walidacja integralności urządzeń, kontrola dostępu, poufność i integralność danych.

Modelując internet rzeczy zgodnie z UML, można wyróżnić przypadki stosowania (ang. *use case*) odpowiadające pojedynczym znaczącym pracom w modelowanym systemie oraz aktorów (ang. *actors*) reprezentujących zewnętrzne jednostki wchodzące w interakcję z systemem.

Wyróżnia się następujące ogólne przypadki stosowania IoT [13]:

- **pomiary i oddziaływanie** – obejmuje łączenie się z fizycznymi rzeczami, pomiar ich stanu oraz oddziaływanie na te rzeczy,
- **zapewnienie usług** – tu przypada działalność związana z dostarczaniem usług i korzystaniem z nich przez użytkowników,
- **zarządzanie danymi** – zbieranie, przesyłanie, przechowywanie i przetwarzanie danych związanych z fizycznymi rzeczami,
- **ochrona prywatności** – zapewnienie zabezpieczenia i ukrycia prywatnych danych pochodzących od fizycznych rzeczy.

Aktorami internetu rzeczy są (w nawiasach podano odpowiadające poszczególnym aktorom role biznesowe z rekomendacji ITU-T Y.2060):

- **fizyczna rzecz,**
- **menedżer danych (dostawca aplikacji lub dostawca urządzenia, zależnie od tego, która z rzeczy ma funkcjonalności związane z zarządzaniem danymi),**
- **dostawca usługi (dostawca aplikacji, platformy, sieci),**
- **użytkownik IoT (użytkownik aplikacji).**

Lista ogólnych wymagań dla internetu rzeczy. Ogólne wymagania dla internetu rzeczy można podzielić na następujące kategorie:

- **niefunkcjonalne:** interoperacyjność, skalowalność, niezawodność, wysoka dostępność, adaptacja, zarządzanie),
- **wsparcie aplikacji:** standardowe programowalne funkcje publiczne, zarządzanie grupami, niezawodna synchronizacja czasu, współpraca urządzeń i usług, by osiągnąć wspólny cel, zarządzanie użytkownikami, gospodarowanie zasobami,

- **usługi:** przydział priorytetów usługom, usługi kontekstowe, komponowanie (składanie) usług, usługi mobilne, wysokiej niezawodności i bezpieczeństwa usługi związane z ciałem człowieka, autonomiczne usługi, usługi oparte na lokalizacji oraz świadome kontekstu, zarządzanie usługami, usługi odkrywania zasobów i użytkowników, zapisywanie się do usług, standaryzowane nazewnictwo i adresacja, wirtualne przechowywanie i moc obliczeniowa,
- **komunikacja:** wsparcie komunikacji opartej na zdarzeniach, cyklicznej oraz automatycznej, wsparcie trybów unicast, multicast, broadcast i anycast, wsparcie komunikacji inicjowanej przez urządzenia, kontrola błędów w komunikacji, wsparcie komunikacji krytycznej ze względu na czas, autonomiczne sieci, komunikacja świadoma treści, komunikacja oparta na lokalizacji, wsparcie różnorodnych technologii komunikacyjnych związanych z urządzeniami, wsparcie różnorodnych technologii komunikacyjnych związanych z siecią,
- **urządzenie:** komunikacja między urządzeniami a internetem rzeczy oparta na identyfikacji, zdalna kontrola, konfiguracja i monitoring urządzeń, zdolność plug and play, terminowy/czasowy monitoring urządzeń, mobilność, kontrola integralności,
- **zarządzanie danymi:** przechowywanie danych, przetwarzanie danych, odpytywanie danych historycznych, kontrola dostępu do danych przez ich właścicieli (właściciel danych kontroluje udostępnianie danych innym), wymiana danych z jednostkami spoza IoT, kontrola integralności oraz zarządzanie cyklem życia danych, znaczeniowe opisywanie oraz dostęp do danych, znaczeniowe przechowywanie, przesyłanie i zbieranie danych,
- **bezpieczeństwo i ochrona danych:** bezpieczeństwo komunikacji, bezpieczeństwo zarządzania danymi, bezpieczeństwo dostarczania usług, integracja różnych polityk i technik bezpieczeństwa, wzajemna autoryzacja i uwierzytelnianie, audyt bezpieczeństwa.

3.2. Znane aplikacje

W procesie projektowania oraz wykonywania niskobudżetowego systemu do ogrzewania energią słoneczną z wymuszoną cyrkulacją zastosowano podejście DIY (ang. *do it yourself* – zrób to sam) polegające na zbudowaniu własnego urządzenia z dostępnych na rynku podzespołów i modułów zamiast zakupu gotowego [14]. Podkreślono przy tym znaczenie filozofii otwartego oprogramowania (ang. *open-source*), która pozwala na znacznie łatwiejsze dostosowanie dostępnego sprzętu i oprogramowania do własnych potrzeb. Wspomniany system składa się z dwóch części: pierwsza jest zasilana energią słoneczną, umieszczona na zewnątrz budynku, w pobliżu kolektora słonecznego i ma za zadanie zbieranie danych pomiarowych z zainstalowanych tam czujników. Druga natomiast znajduje się wewnątrz i odpowiada za pomiar temperatury wody w zbiorniku magazynującym oraz za sterowanie pompami wymuszającymi obieg. Wymiana danych między poszczególnymi częściami systemu została zrealizowana w oparciu o moduły radiowe ZigBee (komunikacja między częścią zewnętrzną i wewnętrzną systemu) oraz protokół Bluetooth (komunikacja z urządzeniami mobilnymi). Zebrane dane są prezentowane na dowolnym urządzeniu mobilnym działającym pod kontrolą systemu Android oraz przechowywane na serwerze w chmurze.

Na uwagę zasługuje MIT App Inventor [15] wykorzystany do przygotowania aplikacji prezentującej dane pomiarowe na urządzeniach mobilnych. Jest to narzędzie oparte na chmurze, które obsługuje się za pomocą przeglądarki internetowej. Wcześniej należy się zalogować z użyciem konta Google. Aplikację się tworzy, korzystając z graficznego języka programowania, natomiast sam proces przypomina układanie puzzli. W trakcie pracy rezultaty można podglądać na posiadanym urządzeniu z systemem Android lub za pomocą symulatora.

W bezprzewodowej sieci czujników oraz inteligentnym domu opartym na IoT do monitorowania stanu zdrowia i samopoczucia mieszkającej w nim osoby zastosowano moduł kontrolujący urządzenia elektryczne i moduły pomiarowe z czujnikami: siły, kontaktu (informacja 0/1), temperatury oraz ruchu (detektory PIR) [16]. Każdy z modułów został wyposażony w przetwornik ZigBee, co umożliwiło zbudowanie bezprzewodowej sieci czujników. Jest ona wykorzystywana do kontroli użytkowania przedmiotów znajdujących się w budynku. Opomiarowane zostały między innymi: stół, krzesło, łóżko, lodówka, czajnik, telewizor, toaleta. Dane pomiarowe są zbierane przez domowy serwer (również wyposażony w ZigBee). Analizując zebrane dane, system wyciąga wnioski dotyczące stanu mieszkańca. Zbadano też problem zakłóceń transmisji radiowej występujących w budynku.

Na podstawie przeprowadzonych doświadczeń sugeruje się wykonanie odpowiednich pomiarów przed instalacją systemu i na tej podstawie zdecydowanie, na którym paśmie występują najmniejsze zakłócenia. Sprzęt radiowy należy ustawić tak, by korzystał z kanału odpowiadającego temu zakresowi częstotliwości.

Badano także zastosowanie sprzętu z otwartym oprogramowaniem (*open-source*) i o otwartej architekturze (*open-hardware*) w bezprzewodowych sieciach czujników [17]. Na poziomie rzeczy elementy i bloki konstrukcyjne IoT są takie same jak w bezprzewodowych sieciach czujników (WSN – *Wireless Sensor Network*). Urządzenia komercyjne często są drogie i własnościowe (zamknięte), co ogranicza dostęp do technologii i spowalnia rozwój. Ich przeciwieństwem są urządzenia typu *open source* i gotowe moduły, o niezastrzeżonej budowie, z których można szybko i łatwo zestawić własne węzły sieci oraz bramy. Pozwala to na budowanie inteligentnych urządzeń i sieci czujników przez większą liczbę osób, obniża bariery dostępu do technologii i przyspiesza rozwój aplikacji IoT. Ponadto projektowanie urządzeń z myślą o szerszej społeczności powinno zmniejszyć problemy z kompatybilnością. Technologie takie jak Raspberry Pi, Beaglebone czy Arduino łatwo rozbudować za pomocą odpowiednich modułów rozszerzających, co pozwoli na tworzenie łatwych w oprogramowaniu i odpornych na błędy aplikacji. Otwartość projektów doprowadziła jednak do nadużyć, dlatego przed zastosowaniem czegokolwiek otwartego do tzw. wysoko wrażliwej aplikacji należy przeprowadzić dokładne testy sprzętu i/lub oprogramowania. W ramach prac zbudowano własne urządzenie z tego typu podzespołów, a następnie porównano je z komercyjnym, zaprojektowanym do określonego celu. Pod uwagę wzięto następujące parametry: wydajność, moc (zużywana energia), koszt urządzenia, zasoby komunikacyjne i obliczeniowe. Wyniki testów pokazały, że w oparciu o otwarte technologie można szybko i łatwo zbudować zaawansowane urządzenie wbudowane o właściwościach porównywalnych lub lepszych od urządzenia komercyjnego zaprojektowanego z myślą o konkretnym zadaniu.

W innej pracy przedstawia się szereg obserwacji związanych z tematyką internetu rzeczy [18]. IoT przykuwa uwagę konstruktorów urządzeń elektronicznych, którzy prześcigają się w różnych rozwiązaniach, ale dostrzegają tu również spory potencjał finansowy. Biorą w tym udział nawet producenci „specjalnych” złączy do aplikacji IoT. Prowadzi się rozważania mające na celu znalezienie odpowiedzi na pytanie: Czym jest IoT? Na początku IoT było bardzo ogólnym hasłem i dotyczyło urządzeń wyposażonych w techniczną możliwość dołączenia się do Internetu. W tym czasie nie do końca potrafiąco też wyjaśnić, w jakim celu należało je dołączać do Internetu. W niektórych przypadkach tak jest do dziś. Zaletą powyższej idei jest coraz większe zapotrzebowanie na urządzenia nowej generacji, które tylko w niewielkim stopniu konkurują z dotychczasowymi rozwiązaniami. To z kolei stwarza szansę na kolejne zyski i nie przeszkadza specjalistom od marketingu w poszukiwaniu pomysłów na zarabianie. Jako przykład takiego działania przytacza się rozwinięcie akronimu IoT (luty 2015 – norymberskie targi Embedded) przez firmę Freescale „Internet of Tomorrow”. Przedstawiciele tej firmy stwierdzili, że posiadają podzespoły i pomysły aplikacyjne dla nadchodzącego internetu jutra, jednak ostatecznie nie zaprezentowali niczego specjalnego. Zwraca się uwagę na potrzebę ustandaryzowania komunikacji urządzeń IoT. W tym celu pracuje już kilka konsorcjów Special Interest Group (SIG), które dążą do opracowania własnych standardów IoT. Obejmują one: specyfikacje grup aplikacyjnych urządzeń, protokoły komunikacyjne, metody ochrony przesyłanych danych przed nieprawnym ich przechwyceniem, definicje klas urządzeń wykorzystujących różne pasma radiowe.

Komitet IEEE stanowi najsilniejsze konsorcjum standaryzujące zaangażowane w przygotowanie specyfikacji IoT. Przez niego został utworzony Global Standards Initiative on Internet of Things (IoT-GSI), który skupia ważne postaci świata Internetu. Grupa IEEE P2413 od końca 2014 roku pracuje nad uelastycznieniem standardów dla różnego rodzaju urządzeń nadających się do podłączenia do Internetu. Tutaj szczególną uwagę obejmuje się integrację pojęć IoT i SmartCity. Członkowie IEEE P2413 współpracują przy tworzeniu standardu z członkami innych grup roboczych: IEEE Communications Society – Power Line Communications (COM/PLC), IEEE Consumer Electronics Society – Standards Committee (CES/SC) oraz IEEE Computer Society – Cloud Computing Standards Committee (C/CCSC). Z ostatnią grupą współpraca ukierunkowana została na integrację wymiany danych przez urządzenia IoT, która ma następować w chmurach obliczeniowych.

Zaznacza się, że IoT jest tak rozpowszechnionym zagadnieniem, iż zainteresowali się nim również członkowie Parlamentu Europejskiego (luty 2015 – European Internet Forum – EIF). Podaje się również alternatywę dla IEEE P2413, którą ma stanowić standard IIoT – Industrial Internet of Things rozwijany przez Object Management Group

pod kątem aplikacji przemysłowych Industry 4.0 oraz otwarty standard IoT, który wykorzystuje w aplikacjach IoT pasma radiowe poniżej 1 GHz. Właśnie ono wzbudza największy niepokój, ponieważ nie ma możliwości zastosowania takich zabezpieczeń, których nie można by było pokonać.

Podaje się jeszcze jeden aspekt związany z internetem rzeczy: wdrażanie IoT rodzi wątpliwości, gdyż jest związane z opłatami w wszystkich dziedzinach życia systemami monitorującymi, na które człowiek nie ma wielkiego wpływu. Pojawia się obawa przed totalitaryzmem – monitorowaniem, a nawet kontrolowaniem zachowania użytkowników urządzeń.

W innej pracy omawia się przemysłowy internet rzeczy będący przemysłową odmianą technologii IoT [19]. Podaje się, że w fabrykach montuje się urządzenia mające na celu gromadzenie danych procesów produkcyjnych i przesyłanie ich do centrów przetwarzania danych. Otrzymane dane służą do optymalizowania działań linii produkcyjnych, a w efekcie do poprawy efektywności procesu wytwarzania. Przedstawiono termin Przemysł 4.0. Tą nazwą określa się proces czwartej rewolucji przemysłowej, w którą wpisuje się technologia przemysłowego internetu rzeczy. Przybliży się również wcześniejsze dokonania:

- **pierwsza rewolucja przemysłowa** – zastosowanie w produkcji urządzeń mechanicznych napędzanych wodą i parą (1784),
- **druga rewolucja przemysłowa** – wprowadzenie podziału pracy i produkcji napędzanej energią elektryczną (linie taśmowe, 1870),
- **trzecia rewolucja przemysłowa** – wprowadzenie do produkcji automatyzacji (sterowniki PLC – 1968 oraz roboty),
- **czwarta rewolucja przemysłowa** – w pełni zautomatyzowane fabryki sterowane przez systemy cybernetyczne.

Ostatnia z wymienionych rewolucji pozwala stale monitorować parametry produkcji, a w razie potrzeby dokonywać zmian w czasie rzeczywistym. Dzięki temu można szybko wykrywać źródła potencjalnych awarii. Również zgromadzone w procesie produkcji dane można zintegrować z oprogramowaniem do ERP (planowanie zasobów przedsiębiorstwa) i CRM (obsługa relacji z klientami). Wtedy produkcja może być dostosowana do zapotrzebowania rynku (produkcja sprzężona z wynikami sprzedaży).

Coraz częściej prognozuje się, że do 2020 roku w prawie połowie procesów w firmach będzie stosowało się automatyczne procesy gromadzenia danych, a w jednej czwartej będą dostępne funkcje autokorekty.

Z internetem rzeczy wiąże się też szereg niebezpieczeństw [20]:

- **łatwość rozpracowania i znalezienia słabych punktów oprogramowania wbudowanego**,
- **możliwość atakowania kolejnych urządzeń przez sieć Internet po „przeniknięciu” do jednego z węzłów sieci**,
- **ignorowanie przez konstruktorów urządzeń kwestii bezpieczeństwa lub brak wiedzy w tym zakresie**,
- **aktualizacja oprogramowania**,
- **duża wrażliwość aplikacji z mocno ograniczonymi zasobami sprzętowymi na złośliwe ataki**,
- **utożsamianie szyfrowania danych z bezpieczeństwem**,
- **możliwość włamania się do sieci ze strony nieautoryzowanego urządzenia**,
- **prostota sieci i związane z tym lekceważenie zabezpieczeń przez użytkowników**,
- **możliwość nieautoryzowanego dołączenia się do sieci czyni szyfrowanie komunikacji nieskutecznym zabezpieczeniem**.

Jako przykład podaje się eksperyment polegający na włamaniu się do systemu informatycznego samochodu Jeep Cherokee. Po uzyskaniu przez badaczy dostępu do magistrali CAN przejęto kontrolę m.in. nad kierownicą, hamulcami, klimatyzacją i wycieraczkami. Kolejnym przykładem było znalezienie luk w bezpieczeństwie pomp infuzyjnych podłączonych do Internetu oraz możliwość zdalnej zmiany celu strzału w karabinie.

W związku z powyższym podaje się trzy filary bezpieczeństwa aplikacji IoT:

- **poufność danych**,
- **integralność**,
- **identyfikacja**.

4. SPRZĘT

Na podstawie przeanalizowanej literatury postanawia się zbadać stosowalność koncepcji internetu rzeczy do elektrod pH przez próbę zbudowania rozproszonego systemu pomiarowego korzystającego z łączności przez Internet. System ten ma zawierać pehametr w roli inteligentnego przetwornika pomiarowego. Poniżej przedstawia się założenia projektowe:

- jonometr w roli urządzenia pomiarowego,
- łączność przez Internet – system rozproszony,
- system SCADA do zbierania danych i kontrolowania urządzeń wchodzących w skład systemu pomiarowego,
- dostęp do danych przez przeglądarkę internetową,
- niskie zużycie energii przez serwer.

Ten inteligentny przetwornik pomiarowy będzie pełnił rolę rzeczy (urządzenia). Dodatkowo znajdzie się komputer, z programem konwertującym protokoły, pełniący rolę bramy sieciowej. Chmura zaś będzie zrealizowana, wykorzystując oprogramowania SCADA.

4.1. Wybór platformy

Na podstawie zestawienia Comparison of single-board computers [8] oraz stron producentów [21, 22] wybrano kilka komputerów jednopłytkowych do dalszego porównania. Kierowano się przy tym następującymi kryteriami:

- moc obliczeniowa (procesor – częstotliwość taktowania, liczba rdzeni, pamięć RAM),
- biblioteki,
- system operacyjny (czy jest wspierany Linux),
- dostępność w kraju,
- cena.

Tab. 1. Porównanie parametrów wybranych komputerów jednopłytkowych

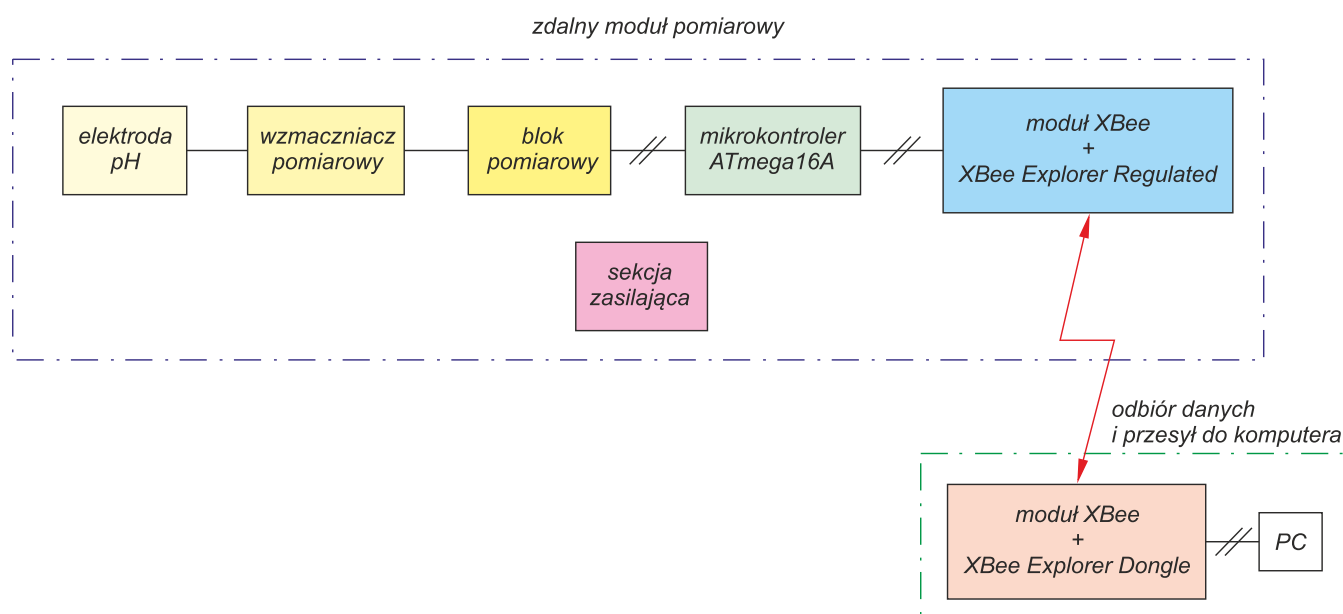
Table 1. Comparison of parameters of selected single-board computers

| Nazwa | Processor | | Pamięć RAM [GB] | USB szt. x typ | Ethernet | Wi-Fi | HDMI | System operacyjny |
|-----------------------|----------------------|--------------------------------|-----------------|--------------------|----------|-------|------|------------------------|
| | Liczba rdzeni [szt.] | Częstotliwość taktowania [GHz] | | | | | | |
| Banana Pi M3 | 8 | 1,8 | 2 | 2 x 2.0 | + | + | + | Linux, Android |
| Cubieboard 4 / CC-A80 | 8 | 1,3 | 2 | 2 x 2.0 | + | - | + | Linux, Android, BSD |
| MarsBoard RK3066 | 2 | 1,6 | 1-2 | 4 x 2.0 | + | + | + | Linux, Android |
| Orange Pi 2 | 4 | 1,6 | 1 | 4 x 2.0 | + | + | + | Linux, Android |
| Orange Pi Mini 2 | 4 | 1,6 | 1 | 4 x 2.0 | + | - | + | Linux, Android |
| Orange Pi PC | 4 | 1,6 | 1 | 3 x 2.0 | + | - | + | Linux, Android |
| Orange Pi Plus | 4 | 1,6 | 1 | 4 x 2.0 | + | + | + | Linux, Android |
| Orange Pi Plus 2 | 4 | 1,6 | 2 | 4 x 2.0 | + | + | + | Linux, Android |
| ODROID C1/C1+ | 4 | 1,5 | 1 | 4 x 2.0 | + | - | + | Linux, Android, NetBSD |
| ODROID-U3 | 4 | 1,7 | 2 | 3 x 2.0 | + | - | + | Linux, Android, NetBSD |
| ODROID-XU | 8 (4+4) | 1,7/1,2 | 2 | ? | ? | ? | ? | Linux, Android, NetBSD |
| ODROID-XU4 | 8 (4+4) | 2,0/1,4 | 2 | 1 x 2.0 2 x 3.0 | + | - | + | Linux, Android, NetBSD |
| ODROID-XU3 Lite | 8 (4+4) | 1,8/1,3 | 2 | 4 x 2.0 1 x 3.0 | + | - | + | Linux, Android, NetBSD |

Wyniki przedstawiono w tabeli 1. Następnie sprawdzono jakość udostępnianej przez producenta dokumentacji, dostępność oprogramowania (obrazy systemu Linux, biblioteki) oraz zapoznano się z opinią osoby, która testowała wybrane układy [23, 24, 25]. Na tej podstawie podjęto wybór platformy sprzętowej, który padł na komputer jednopłytkowy ODROID C1+ ze względu na spełnienie minimalnych wymagań (pamięć, moc obliczeniowa), odpowiedni stosunek parametrów do ceny, dobra opinia oraz dostępność.

4.2. Jonometr

Jako jonometr wykorzystano układ zaprojektowany, zmontowany i uruchomiony wcześniej [26]. Układ ten, przedstawiony na rysunku 1, składa się z następujących bloków funkcjonalnych: (1) wzmacniacza pomiarowego INA116 – zapewnia odpowiednio wysoką impedancję wejściową, która pozwala na wykonanie bezprądowego pomiaru wartości napięcia na elektrodzie pH; (2) bloku pomiaru napięcia – 22-bitowego przetwornika analogowo-cyfrowego MCP3551-E/SN wraz z elementami pomocniczymi; (3) mikrokontrolera ATmega16A – sterowanie całym układem i obsługa komunikacji; (4) modułu XBee – moduł radiowy pracujący w standardzie ZigBee zapewniający bezprzewodowe przesyłanie danych; (5) przejściówki XBee Explorer Regulated – zapewnia zasilanie modułu XBee napięciem +3,3 V oraz dopasowuje poziomy napięć sygnałów wymienianych z mikrokontrolerem; (6) sekcji zasilania – zapewnienie odpowiednich napięć pozostałym blokom układu.



Rys. 1. Schemat blokowy układu pomiarowego poddanego modyfikacjom [26]

Fig. 1. Block diagram of measurement system subject to modifications [26]

4.3. Układy przetwarzania standardów komunikacyjnych

Moduł elektroniczny między jonometrem a systemem nadrzędnym składa się z dwóch przetworników: pierwszy jest przyłączony do jonometru i dokonuje przetwarzania sygnału z UART na standard RS232, natomiast drugi służy do przejścia z RS232 na USB. Przetwornik UART-RS232 jest typowym rozwiązaniem korzystającym z układu MAX232. Został podłączony do jonometru w miejsce modułu ZigBee wykorzystywanego wcześniej, z tą różnicą że otrzymuje zasilanie ciągłe (w poprzedniej konfiguracji sekcja komunikacyjna była włączana i wyłączana razem z sekcją pomiarową). Jest to konieczne do wymiany danych z systemem SCADA za pośrednictwem protokołu Modbus.

5. OPROGRAMOWANIE

5.1. Przygotowanie oprogramowania

Prace przeprowadzono pod systemem operacyjnym Linux. Rozpoczęto od pobrania ze strony producenta odpowiedniego obrazu systemu Ubuntu Linux w wersji 14.04lts dla posiadanego komputera jednopłytkowego odroid.in. Następnie obraz ten rozpakowano. Kolejnym krokiem było nagranie obrazu na kartę pamięci.

System OpenSCADA nie występuje w postaci gotowego pakietu debianowego dla architektury armhf, na której opiera się komputer jednopłytkowy Odroid C1+. Z tego powodu wskazane jest przygotowanie pakietu we własnym zakresie przy wykorzystaniu wskazówek zawartych w dokumentacji programu [27]. Należy pobrać pakiety źródłowe **opencada** oraz **opencada-res** z serwera ftp projektu. Konieczne jest rozwiązanie zależności zgodnie z wymaganiami [28] poprzez doinstalowanie brakujących pakietów oraz wykonanie konfiguracji za pomocą polecenia **./configure-enable-AllModuls**. Wybrana opcja oznacza wykonanie konfiguracji przy włączeniu do budowanego programu wszystkich modułów. W pracy konieczne było zainstalowane następujących pakietów: `zlib1g-dev`, `libmysqld-dev`, `firebird-dev`, `libpq-dev`, `libsnp-dev`, `libportaudiocpp0`, `portaudio19-dev`, `libqt4-dev`, `libgd-dev`, `libphonon-dev`, `phonon-backend-gstreamer`, `phonon-backend-gstreamer1.0`. Budowanie pakietu uruchomiono poleceniem: **dpkg-buildpackage-rfakeroot**. Opcja **-rfakeroot** oznacza, że w przypadku działań wymagających uprawnień roota zostanie użyte polecenie **fakeroot**. Zapewnia ono aplikacji środowisko działania, które sprawia „wrażenie” posiadania uprawnień roota do manipulacji plikami bez ich faktycznego uzyskania [29, 30]. Konieczne okazało się również doinstalowanie dodatkowych pakietów: `libphonon-dev`, `libphonon4qt5-dev`. Wystąpiły także problemy ze zgodnością wersji przy budowaniu modułu odpowiadającego za urządzenia firmy ICP DAS. Moduł ten nie jest potrzebny, więc został wyłączony przez usunięcie w pliku `/debian/rules` wpisu **--enable-ICP_DAS** z opcji polecenia **configure**.

5.2. Zastosowanie protokołu Modbus RTU

W celu umożliwienia komunikacji z przetwornikiem pomiarowym zaimplementowano protokół Modbus RTU. Wykorzystano przy tym dostępne informacje [8: Modbus, 31]. Do wyliczenia sum kontrolnych CRC16 zastosowano bibliotekę **util/crc16.h**, natomiast do deklarowania zmiennych o ściśle określonej liczbie bitów wykorzystano **stdint.h**. Program w języku C również przygotowano jako bibliotekę, która obejmuje podstawowe kody funkcji protokołu. Podstawową funkcją we wspomnianej bibliotece jest **ModbusRTU()**. Jako parametry funkcja przyjmuje wskaźnik na tablicę (bufor) z odebranymi danymi, indeks ostatniego bajtu w tablicy wejściowej, wskaźnik na tablicę, w której należy zapisać odpowiedź, wskaźniki na odpowiednie rejestry modbusowe oraz maksymalne wartości adresów modbusowych. Wartością zwracaną jest indeks ostatniego elementu zapisanego do tablicy wyjściowej. Na początku, za pomocą funkcji **PoliczCRC()**, wyliczana jest suma kontrolna CRC16 otrzymanych danych. Następnie jest ona porównywana z wartością otrzymaną w odebranym komunikacie. Jeśli CRC jest zgodne, wykonywane są dalsze polecenia. W innym przypadku funkcja zwraca zero.

Instrukcja **switch()** sprawdza, która funkcja modbusowa ma zostać wykonana, i wywołuje odpowiednią komendę z biblioteki. Jeśli żadna z dostępnych funkcji nie zostanie rozpoznana, zgłaszany jest wyjątek 1 (nielegalna funkcja – slave nie rozpoznaje danej funkcji lub nie jest ona dozwolona). Na końcu wyliczana jest suma kontrolna wygenerowanej odpowiedzi. Wartość CRC jest dołączana w ostatnich dwóch bajtach, przy zachowaniu zasady, że najmniej znaczący bajt jest wpisywany (i wysyłany) jako pierwszy. W celu zamiany liczby szesnastobitowej na dwie ośmiobitowe komórki w tablicy zastosowano odpowiednie przesunięcia bitowe.

Funkcja 01 i 02: Odczyt cewek i odczyt wejść cyfrowych

Zapytanie:

| | |
|---|---------------|
| Zawartość | Długość, bity |
| Adres pierwszej cewki lub wejścia cyfrowego do odczytania | 16 |
| Liczba cewek lub wejść cyfrowych do odczytania | 16 |

Odpowiedź:

| | |
|---|---------------|
| Zawartość | Długość, bity |
| Liczba następujących bajtów z odczytanymi wartościami | 8 |
| Odczytane wartości cewek lub wejść cyfrowych | 8 na bajt |

Pierwsza z cewek (lub wejście cyfrowe) jest przechowywana jako LSB pierwszego bajtu odpowiedzi. Jeśli liczba odczytywanych cewek (wejść cyfrowych) nie jest wielokrotnością ósemki, to MSB ostatniego bajtu są uzupełniane zerami. Funkcja przyjmuje wskaźnik na tablicę z danymi wejściowymi, wskaźnik na tablicę, do której należy zapisać

wynik działania funkcji, wskaźnik na rejestr cewek, maksymalny adres cewki oraz wskaźnik, przez który należy zwrócić pozycję ostatniego zapisanego bajtu. W pierwszej kolejności z otrzymanych bajtów odtwarzane są wartości szesnastobitowe (adres pierwszej cewki i liczba cewek do odczytania). Jest to realizowane przez odpowiednie przesunięcia bitowe: najmniej znaczący bajt jest wpisywany do zmiennej, a następnie najbardziej znaczący bajt jest przesuwany o 8 miejsc w lewo i sumowany logicznie ze zmienną przechowującą wartość szesnastobitową. Następnie sprawdza się, czy nie została przekroczona maksymalna wartość adresu modbusowego i w razie potrzeby zgłaszany jest odpowiedni wyjątek (nielegalny adres danych, co odpowiada sytuacji, w której część lub wszystkie adresy danych nie są dozwolone lub nie istnieją w danym urządzeniu slave). Jeśli adresy żądanych rejestrów mieszczą się w dopuszczalnych granicach, do tablicy wyjściowej wpisywana jest liczba bajtów z wartościami, które nastąpią, a następnie wpisuje się wartości odpowiednich cewek. Licznik jest ustawiany tak, by wskazywał ostatni zapisany bajt bufora. Kończąc działanie, funkcja zwraca zero. Analogicznie została oprogramowana funkcja 02, z tym, że działa na rejestrze wejść cyfrowych.

Funkcja 03 i 04: Odczyt rejestrów przechowujących i odczyt rejestrów wejściowych

Zapytanie:

| | |
|--|---------------|
| Zawartość | Długość, bity |
| Adres pierwszego z rejestrów do odczytania | 16 |
| Liczba rejestrów do odczytania | 16 |

Odpowiedź:

| | |
|---|---------------------|
| Zawartość | Długość, bity |
| Liczba następujących bajtów z odczytanymi wartościami rejestrów | 8 |
| Wartości rejestrów | 16 na jeden rejestr |

Funkcja przyjmuje jako argument wskaźnik na tablicę z odebranymi danymi, wskaźnik na tablicę, do której należy zapisać rezultaty działania funkcji, wskaźnik na rejestry przechowujące, maksymalny adres rejestru przechowującego oraz wskaźnik na licznik, który należy ustawić tak, by zawierał indeks ostatniego zapisanego do tablicy wyjściowej bajtu. Pierwszą operacją jest odtworzenie wartości szesnastobitowych (adresu pierwszego rejestru oraz liczby rejestrów do odczytania). Następnie kontroluje się, czy adres ostatniego z żądanych rejestrów nie przekracza maksymalnego dopuszczalnego dla danego urządzenia (jeśli trzeba zgłaszany jest odpowiedni kod wyjątku). Kolejnym krokiem jest sprawdzenie, czy liczba żądanych rejestrów nie przekracza maksymalnej liczby, którą można odczytać za jednym razem. Ograniczenie wynika z faktu, że pole przechowujące liczbę bajtów z wartościami w odpowiedzi ma jeden bajt długości. W razie przekroczenia wspomnianego ograniczenia liczba odczytywanych rejestrów zostaje odpowiednio zmniejszona. Wyznaczana jest liczba bajtów z wartościami rejestrów w odpowiedzi. Wartości odpowiednich rejestrów są wpisywane do tablicy wyjściowej. Wymaga to zamiany wartości szesnastobitowych na dwie ośmiobitowe, co jest realizowane przez odpowiednie operacje bitowe. Liczba szesnastobitowa jest przesuwana o 8 miejsc w lewo, a następnie o 8 miejsc w prawo, co pozwala wyłuskać z niej najmniej znaczące bity. Wynik operacji jest wpisywany do tablicy elementów ośmiobitowych. Najbardziej znaczące bity uzyskuje się przez przesunięcie liczby szesnastobitowej o 8 miejsc w prawo i wpisanie wartości do zmiennej ośmiobitowej. Na końcu ustawiana jest wartość licznika zawierającego indeks ostatniego bajtu zapisanego do tablicy wyjściowej. Funkcja zwraca zero.

Funkcja 05: Zapis pojedynczej cewki

Zapytanie:

| | |
|----------------------|---------------|
| Zawartość | Długość, bity |
| Adres cewki | 16 |
| Wartość do zapisania | 16 |

Odpowiedź: identyczna z zapytaniem

Jeśli cewka ma zostać wyłączona w polu wartość jest zero, natomiast jeśli cewka ma zostać włączona to pole wartość zawiera 0xFF00 (65280). Argumentami funkcji są: wskaźnik na tablicę z odebranymi bajtami, wskaźnik na tablicę, w której należy umieścić odpowiedź, wskaźnik na rejestr cewek, maksymalny adres cewki oraz wskaźnik na licznik położenia ostatniego zapisanego elementu. Na początku odtwarzane są wartości szesnastobitowe. Później następuje sprawdzenie, czy adres z zapytania mieści się w dozwolonym zakresie. Jeśli tak nie jest, zgłoszony zostaje odpowiedni kod wyjątku. Sprawdzana jest również wartość zapisywana do cewki. Jeśli nie odpowiada jednej z dopuszczalnych (0 lub 0xFF00), zgłaszany jest kod wyjątku nielegalna wartość danych. Ustawiany jest licznik położenia ostatniego bajtu zapisanego do tablicy wyjściowej. Funkcja zwraca zero.

Funkcja 06: Zapis pojedynczego rejestru pamiętającego

Zapytanie:

| Zawartość | Długość, bity |
|---|---------------|
| Adres rejestru pamiętającego do zapisania | 16 |
| Nowa wartość rejestru pamiętającego | 16 |

Odpowiedź: identyczna z zapytaniem

Funkcja przyjmuje jako argumenty: wskaźnik na tablicę z odebranymi bajtami, wskaźnik na tablicę, w której należy umieścić odpowiedź, wskaźnik na rejestry pamiętające, maksymalny adres rejestru pamiętającego oraz wskaźnik na licznik położenia ostatniego zapisanego elementu. Pierwszą operacją jest odtworzenie szesnastobitowego adresu rejestru. Następnie sprawdza się, czy adres ten mieści się w dopuszczalnym zakresie (jeśli nie, to zgłaszany jest odpowiedni kod wyjątku). Odtwarzana jest szesnastobitowa wartość, po czym jest ona wpisywana do rejestru wskazywanego przez adres. W kolejnym kroku wpisuje się odpowiedź do tablicy wyjściowej. Działanie funkcji kończy ustawienie licznika położenia ostatniego bajtu i zwrócenie zera.

Funkcja 15: Zapis wielu cewek

Zapytanie:

| Zawartość | Długość, bity |
|---|---------------|
| Adres pierwszej cewki do zapisania | 16 |
| Liczba cewek do zapisania | 16 |
| Liczba następujących bajtów z wartościami cewek | 8 |
| Wartości cewek | 8 na bajt |

Wartości cewek są podawane binarnie: 0 dla wyłączonej i 1 dla włączonej. Wartość pierwszej z cewek jest przechowywana w LSB pierwszego bajtu z wartościami. Jeśli liczba cewek nie jest wielokrotnością ósemki, to MSB ostatniego bajtu są uzupełniane zerami.

Odpowiedź:

| Zawartość | Długość, bity |
|-----------------------|---------------|
| Adres pierwszej cewki | 16 |
| Liczba cewek | 16 |

Funkcja przyjmuje jako argument wskaźnik na tablicę z odebranymi bajtami, wskaźnik na tablicę, w której należy umieścić odpowiedź, wskaźnik na rejestr cewek, maksymalny adres cewki oraz wskaźnik na licznik położenia ostatniego zapisanego elementu. Odtwarzane są wartości szesnastobitowe. Sprawdza się, czy adres ostatniej cewki mieści się w dopuszczalnym zakresie. Jeśli trzeba, zgłaszany jest odpowiedni kod wyjątku. Nowe wartości cewek zapisywane są do rejestru. Przygotowywana jest odpowiedź. Przedostatnią operacją jest ustawienie licznika zawierającego indeks ostatniego zapisanego bajtu. Funkcja zwraca zero.

Funkcja 16: Zapis wielu rejestrów przechowujących**Zapytanie:**

| | |
|---|---------------------|
| Zawartość | Długość, bity |
| Adres pierwszego z rejestrów przechowujących do zapisania | 16 |
| Liczba rejestrów do zapisania | 16 |
| Liczba następujących bajtów z wartościami do zapisania w rejestrach | 8 |
| Nowe wartości rejestrów przechowujących | 16 na jeden rejestr |

Odpowiedź:

| | |
|---|---------------|
| Zawartość | Długość, bity |
| Adres pierwszego z zapisanych rejestrów przechowujących | 16 |
| Liczba zapisanych rejestrów przechowujących | 16 |

Jako argumenty funkcja przyjmuje: wskaźnik na tablicę z odebranymi bajtami, wskaźnik na tablicę, w której należy umieścić odpowiedź, wskaźnik na rejestry przechowujące, maksymalny adres rejestru przechowującego oraz wskaźnik na licznik położenia ostatniego zapisanego elementu. Na początku odtwarzane są szesnastobitowe wartości – adres pierwszego rejestru oraz liczba rejestrów do zapisania. Jeśli adres ostatniego rejestru przekracza dopuszczalny zakres, zgłaszany jest odpowiedni kod wyjątku. Następuje zapis wartości do rejestrów. Formowana jest odpowiedź. Działanie funkcji kończy ustawienie licznika wyjściowego i zwrócenie zera.

Obliczanie sumy kontrolnej CRC16. Wylizanie sumy kontrolnej CRC16 jest realizowane za pomocą funkcji **PoliczCRC()**. Przyjmuje ona wskaźnik na tablicę zawierającą dane, których sumę kontrolną należy wyznaczyć, oraz długość ciągu bajtów. Wartością zwracaną jest wyznaczona suma CRC16. Na początku wartość CRC ustawiana jest na 0xFFFF. Następnie wywołuje się w pętli funkcję **_crc_update()** z biblioteki **util/crc16.h**, która uaktualnia CRC bajt po bajcie.

Raportowanie błędów. Jeśli slave ma potrzebę zgłoszenia błędu, to w odpowiedzi wysyła kod funkcji zwiększony o 128, a w polu danych zamieszcza odpowiedni kod błędu.

5.3. Jonometr – modyfikacja programu

Rozwiązanie przedstawione w tej pracy opierało się na wcześniejszym układzie [26]. Do najistotniejszych zmiany należy implementacja protokołu Modbus RTU. Dodano odpowiednie rejestry protokołu: jeden rejestr wejściowy (wynik pomiaru – zmierzony potencjał) oraz jeden rejestr przechowujący (wartość okresu próbkowania podana w minutach). Adresy wspomnianych rejestrów wynoszą zero. Dodano również możliwość ustawienia okresu próbkowania zdalnie za pomocą protokołu Modbus RTU. Dopuszczalny zakres wynosi od 1 do 15 minut. Możliwe jest też ustawienie wartości zero, co powoduje wykonywanie jednego pomiaru za drugim.

5.4. Program do przetwarzania protokołów Modbus RTU–Modbus TCP

W celu umożliwienia komunikacji przez Internet napisano program przetwarzający protokoły. Jest to program konsolowy działający pod kontrolą systemu operacyjnego Linux. Idea działania tego programu jest następująca:

1. Odebranie ramki Modbus TCP.
2. Przetwarzania odebranej ramki protokołu Modbus z wersji TCP na RTU (w tym wylizanie sumy kontrolnej CRC16).
3. Przesłanie ramki Modbus RTU do urządzenia slave (sterownika lub przetwornika pomiarowego).
4. Oczekiwanie na odpowiedź zabezpieczone przed awariami typu wyłączenie sterownika przy pomocy limitu czasu (ang. *timeout*).
5. Odebranie odpowiedzi ze sterownika (przetwornika pomiarowego).
6. Sprawdzenie sumy kontrolnej CRC16.
7. Jeśli suma kontrolna nie jest poprawna, zapytanie jest ponownie wysyłane do sterownika (przetwornika pomiarowego).

8. Przetwarzanie ramki protokołu z RTU na TCP.

9. Wysłanie ramki Modbus TCP.

6. SYSTEM OPENSADA

6.1. Aplikacja testowa

W ramach testów oraz w celu zapoznania się z systemem OpenSCADA przygotowano aplikację testową, korzystając z instrukcji QuickStart [32] oraz przykładów i bibliotek gotowych elementów dostępnych we wspomnianym pakiecie. Konieczne było wykonanie następujących etapów:

- dodanie sterownika w module Data Acquisition / Modbus,
- konfiguracja łącza szeregowego (zastąpionego później łączem internetowym) w module Transports,
- dodanie parametru do utworzonego wcześniej obiektu sterownika w module Data Acquisition / Modbus,
- konfiguracja zmiennych, które mają być wymieniane ze sterownikiem,
- przygotowanie wizualizacji.

Do przeprowadzenia testów wykorzystano sterownik oparty na mikrokontrolerze wyposażony w przekaźnik i program do obsługi protokołu Modbus RTU oraz komputer Odroid C1+, który pełnił rolę serwera systemu SCADA. W kolejnym etapie sterownik został zastąpiony przez jonometr ze zmodyfikowanym programem. Przeprowadzone testy wykazały, że urządzenia łączą się ze sobą, a odpowiednie dane są wymieniane.

6.2. Udostępnianie przez Internet

Skonfigurowano system OpenSCADA tak, by można było się z nim połączyć za pośrednictwem przeglądarki internetowej. W tym celu w Visual Control Area Engine dodano sesję oraz ustawiono użytkownika oraz projekt, który miał być udostępniany przez Internet. Sprawdzono również konfigurację modułu WebVision. Upewniono się, że poniższe parametry mają właściwe wartości: life time of session: 10 min, sessions limit: 5, PNG compression level: 1. Konieczne okazało się również ustawienie dla procesu uruchamianego wraz ze startem systemu operacyjnego komputera odpowiedniej bazy danych z konfiguracją. Zanim to wykonano, logując się do systemu SCADA za pośrednictwem przeglądarki internetowej, nie było dostępu do wizualizacji i danych pomiarowych.

Zdalny dostęp do panelu operatorskiego programu jest możliwy pod adresem w formacie: **http://<adres_serwera>:10002**. Po wejściu na tę stronę można wybrać, z którego modułu chce się skorzystać. Do wyboru są między innymi: panel operatorski i konfigurator systemu OpenSCADA w dwóch wersjach. Następnie należy się zalogować z użyciem nazwy użytkownika i hasła.

7. WNIOSKI

Zagadnienie internetu rzeczy jest bardzo obszerne. Do dziś trwają prace mające na celu ustalenie standardów IoT. W wielu publikacjach podawane są ogólne informacje oraz wymagania, które nie pozwalają jednoznacznie określić, czym dokładnie jest **Internet of Things (Internet rzeczy)**. Źródłami, które najlepiej opisują to zagadnienie, są rekomendacje ITU.

W opisywanym systemie pehametr zbudowany w oparciu o wejściowy przetwornik sygnału i typowy układ mikroprocesorowy wyposażono w możliwość komunikacji z użyciem standardu przemysłowego w wersji TCP/IP. Dało to możliwość włączenia go np. w system SCADA bez większych ograniczeń terytorialnych.

Z pewnością można uznać zbudowany przetwornik pomiarowy (pehametr) za rzecz (urządzenie). Ogólnie takich urządzeń mogłoby być więcej i niekoniecznie wszystkie musiałyby być miernikami pH, czy w ogóle urządzeniami pomiarowymi. Mogłoby to być np. urządzenie wykonawcze lub inne, które może być kontrolowane za pomocą protokołu Modbus (tak jak w sieciach przemysłowych). Komputer z programem przetwarzającym wersje protokołu pełni rolę bramy sieciowej. Zapewnia on niezbędną przemianę protokołów i możliwość użycia urządzeń z innym protokołem czy standardem elektronicznym. Z kolei serwer z systemem SCADA może być nazwany substytutem chmury. Dostęp zdalny przez internet pozwala kontrolować pomiary wykonywane przez urządzenia w dowolnym miejscu.

Internet rzeczy jest technologią na tyle uniwersalną, że również taki pehometr, jako reprezentant szerszego grona urządzeń pomiarowych, może z powodzeniem zostać jego częścią. Przeprowadzone badania wykazały również, że cały system pomiarowy jaki znamy obecnie może, po odpowiednim przystosowaniu, stać się częścią IoT jako rozproszony system pomiarowy.

Praca finansowana ze środków Ministerstwa Nauki i Szkolnictwa Wyższego (Bk/213/RAu1/2016).

BIBLIOGRAFIA

- [1] S. Ferber, „Jak internet rzeczy wpływa na naszą rzeczywistość”, https://www.hbrp.pl/b_jak_internet_rzeczy_wplywa_na_nasza_rzeczywistosc/b8nnarhx.
- [2] E. Kwiatkowska, „Rozwój internetu rzeczy – szanse i zagrożenia”, *Internetowy Kwartalnik Antymonopolowy i Regulacyjny*, vol. 3, no. 8, pp. 60–70, 2014. <http://ikar.wz.uw.edu.pl/numery/22/pdf/60.pdf>.
- [3] „Internet rzeczy – czym jest i jakie niesie zagrożenia”, *Komputer Świat*. http://www.komputerswiat.pl/sekcje_specjalne/klikaj_bezpiecznie_zagrozenia/internet_rzeczy_czym_jest_i_jakie_niesie_zagrozenia.aspx.
- [4] J. Piotrowski, „Podstawy miernictwa”, *WNT*, 2002.
- [5] R. Schillak, „Oznaczenie *pH* w glebach”, *Roczn. Glebozn.*, vol. 7, no. dodatek, pp. 26–39, 1958. http://ssa.ptg.sggw.pl/files/artykuly/1958_07_dodatek/tom_7_nr_dodatek_25-39.pdf.
- [6] A. Kozyra, A. Wiora, J. Wiora, „Szacowanie niepewności w pomiarach potencjometrycznych”, *Katowice: PAN, PKJS*, 2007.
- [7] J. Frączek, S. Waluś, eds, „Laboratorium miernictwa przemysłowego”, *Wydawnictwo Politechniki Śląskiej*, 2002.
- [8] Wikipedia. <http://wikipedia.org>.
- [9] Z. Bodnar, Zygmunt Klemensiewicz 1886—1963. http://www.ifpan.edu.pl/ON_1/Historia/art/32kle.pdf.
- [10] K. Maksymiuk, A. Michalska, „Elektrody jonoselektywne – klasyka i nowe koncepcje”, *Chemik*, vol. 69, no. 7, pp. 373–382, 2015. http://miesiecznikchemik.pl/wp_content/uploads/2015/08/chemik_2015_07_1.pdf.
- [11] K. Ashton, „That 'internet of things' thing”, <http://www.rfidjournal.com/articles/view?4986>.
- [12] ITU-T Y.2060. „Overview of the Internet of things”, https://www.itu.int/rec/dologin_pub.asp?lang=e&id=T_REC_Y.2060_201206_1!!PDF_E&type=items.
- [13] ITU-T Y.2066. „Common requirements of the Internet of things”, https://www.itu.int/rec/dologin_pub.asp?lang=e&id=T_REC_Y.2066_201406_1!!PDF_E&type=items.
- [14] F. Salamone, L. Belussi, L. Danza, M. Ghellere, I. Meroni, „An open source low-cost wireless control system for a forced circulation solar plant”, *Sensors*, vol. 15, pp. 27 990–28 004, 2015. http://www.mdpi.com/1424_8220/15/11/27990/htm.
- [15] „Explore MIT App Inventor”, <http://appinventor.mit.edu>.
- [16] WSN, I.-B. S. Homes, and T. E. to Smart Buildings, „Wsn- and iot-based smart homes and their extension to smart buildings”, *Sensors*, no. 15, pp. 10 350–10 379, 2015. http://www.mdpi.com/1424_8220/15/5/10350/htm.
- [17] R. Fisher, L. Ledwaba, G. Hancke, and C. Kruger, „Open hardware: A role to play in wireless sensor networks?”, *Sensors*, vol. 15, pp. 6818–6844, 2015. http://www.mdpi.com/1424_8220/15/3/6818/htm.
- [18] P. Zbysiński, „Iot czyli...?”, *Elektronika Praktyczna*, vol. 2 pp. 71–73, 2016. http://ep.com.pl/artykuly/10918-_IoT_czyli.html.
- [19] L. Krysiewicz, „Przemysłowy internet rzeczy. Mikrokontroler CC1310 i zestaw startowy CC1310 LaunchPad”, *Elektronika Praktyczna*, vol. 10, pp. 84–87, 2016. http://ep.com.pl/artykuly/11297-_Przemyslowy_Internet_Rzeczy_Mikrokontroler_CC_i_zestaw_startowy_CC_LaunchPad.html.
- [20] Z. Kubiak, „Wprowadzenie do internetu przedmiotów”, http://fc.put.poznan.pl/materials/84-_zkubiak-_iot-_wprowadzenie.pdf.
- [21] „ODROID-C1+”, http://www.hardkernel.com/main/products/prdt_info.php?g_code=G143703355573.
- [22] „Orange Pi”, <http://www.orangepi.org>.
- [23] „ODROID-C1 i zapomnij o Raspberry Pi”, [Online]. Available: <http://www.jarzebski.pl/blog/2015/01/21/odroid-c1-i-zapomnij-o-raspberrypi.html>
- [24] „Nadchodzi poprawiona wersja ODROID-C1+”, [Online]. Available: <http://www.jarzebski.pl/blog/2015/07/24/nadchodzi-poprawiona-wersja-odroid-c1.html>
- [25] „Test nowego ośmiordzeniowca – Banana PI M3”, <http://www.jarzebski.pl/blog/2016/01/22/test-nowego-osmiordzeniowca-banana-pi-m3.html>.
- [26] M. Michalski, „Układ transmisji danych pomiarowych standardem ZigBee – projekt inżynierski”, *Politechnika Śląska*, 2015.
- [27] R. Savochenko, „Manual for OpenSCADA building from sources”, <http://wiki.oscada.org/HomePageEn/Doc/BuildFromSource?v=18ee>.
- [28] M. Lysenko, „Functional characteristics and demands of OpenSCADA system”, <http://wiki.oscada.org/HomePageEn/Function>.

[29] "Linux man pages online: dpkg-buildpackage(1)", http://man7.org/linux/man_1/dpkg-buildpackage.1.html.

[30] "Linux Man Pages Online: fakeroot", <http://man.he.net/man1/fakeroot>.

[31] "Simply Modbus. Exception Responses", <http://www.simplymodbus.ca/exceptions.htm>.

[32] R. Savochenko, M. Lysenko, "OpenSCADA Wiki: Quick Start", <http://wiki.oscada.org/HomePageEn/Doc/QuickStart>.

otrzymano / received: 20.09.2016 przyjęto do publikacji / accepted: 20.10.2016