پروژه ۱

عنوان: توسعه یک برنامه سفارشی در کاربرد اینترنت اشیاء	**نوع پروژه:** کاربردی
ایمیل طراح: mahdi.bahreiny96@sharif.edu	**پیش نیاز:** مفاهیم کلی سیستم عامل

توضیحات

مفهوم اینترنت اشیا (IoT) بیانگر ارتباط فناوری‌هایی مانند حسگر، واحد پردازش، نرم‌افزار و غیره با دیگر سیستم‌ها از طریق اینترنت یا دیگر شبکه‌های ارتباطی است. شبکه‌های رایانه‌ای در انواع گسترده اجازه می‌دهند تا اشیاء در سراسر جهان تحت زیرساخت‌های شبکه‌ی موجود، از راه دور کنترل شوند(مانند خانه هوشمند) و همچنین فرصتی برای ادغام مستقیم جهان فیزیکی با سیستم‌های مبتنی بر رایانه را فراهم آورده و علاوه بر کاهش دخالت انسان، منجر به بهبود بهره‌وری، افزایش دقت و بهره اقتصادی شده است (مانند ربات خانگی). استفاده از سیستم های امبدد در کاربرد اینترنت اشیاء به دلایلی مانند مصرف انرژی کم، نیاز به فضای کوچک، کاهش تعداد منابع سخت افزاری سیستم و کاهش هزینه ی ساخت روز به روز در حال افزایش است. همزمان با رشد استفاده از این نوع دستگاه ها، توسعه ی برنامه های سفارشی برای آن ها نیز گسترش یافته است. هدف از این پروژه، توسعه یک برنامه کاربردی سفارشی بر پایه کیوت (Qt) است که قابلیت تعامل با اشیاء هوشمند را داشته باشد. این برنامه به عنوان یک تصویر سفارشی لینوکس ایجاد خواهد شد و قابلیت اجرا بر روی پلتفرم‌های مختلف، از جمله Raspberry Pi را داراست. استفاده از Qt به دلیل قابلیت‌های قوی در ایجاد رابط کاربری گرافیکی (GUI) و توسعه نرم‌افزارهای چندسکویی، انتخاب مناسبی برای این پروژه است.

ابزار مورد نیاز

- بورد رزبری پای
- LCD لمسی
- فریم ورک QT
- ابزار ساخت تصویر سفارشی لینوکس

شرح پروژه

Qt یک Application Framework با قابلیت اجرا در سیستم عامل های مختلف است. با این نرم افزار میتوان نرم افزار هایی را توسعه داد که در پلتفرم های مختلف (هم نرم افزاری هم سخت افزاری) بدون کمترین تغییر در کد قابلیت اجرا داشته باشند. این Framework به طور پیشفرض از زبان های C++ و QML استفاده می کند. در این پروژه در ابتدا یک برنامه با رابط کاربری در پلتفرمQT نوشته می شود. در این برنامه بایستی تعدادی سنسور و عملگر به صورت مجازی تعریف کرد و از طریق اینترنت بتوان با این سیستم ارتباط برقرار کرد. بهتر است این برنامه از همان ابتدا در بورد زربری پای توسعه داده شود.

سنسورهای مجازی شامل سنسور روشنایی، سنسور دما و سنسور تشخیص حرکت می باشند. سنسور روشنایی دارای ۵ سطح روشنایی است(از خیلی تاریک تا خیلی روشن). سنسور دما یک عدد طبیعی بین ۰ تا ۳۰ درجه تولید می کند. سنسور تشخیص حرکت نیز دارای دو وضعیت تشخیص/عدم تشخیص حرکت است. مقادیر تولید شده توسط این سنسورهای مجازی بایستی به صورت تصادفی بوده و آخرین مقدار آن ها بر روی رابط کاربری نمایش داده شود.

عملگرهای مجازی شامل لامپ، پرده برقی و کولر آبی می باشد. عملگرهای لامپ و پرده برقی دارای دو حالت روشن و خاموش بوده و کولر آبی چهار حالت دارد (خاموش، دور کند، دور متوسط، دور تند). آخرین وضعیت این عملگرها روی رابط کاربری نمایش داده می شود و کاربر بایستی بتواند وضعیت آن ها را تغییر دهد.

قابلیت دیگری که این برنامه بایستی داشته باشد، اتصال به اینترنت است. این ارتباط می تواند به صورت سیمی یا بیسیم باشد. این برنامه بایستی به یک سرور متصل شده، داده های سنسورها و عملگرها را به آن بفرستد و همچنین بتواند دستورات لازم برای تغییر وضعیت عملگرها را دریافت نماید. برای برنامه ی سمت سرور داشتن یک رابط کاربری ساده کافی است.

در ادامه، این برنامه و وابستگی های آن با استفاده از ابزار ساخت تصویر سفارشی لینوکس مانند Yocto، Buildroot یا Open Embedded به صورت یک نسخه سفارشی از لینوکس پیاده سازی می شود. بدین صورت که پس از بوت شدن سیستم تنها برنامه ای که توسعه داده شد، اجرا شود.

در انتها این برنامه بایستی روی بورد رزبری پای با استفاده از LCD لمسی اجرا شود.

مراحل انجام پروژه

- توسعه برنامه اصلی با استفاده از فریم ورک QT
- توسعه برنامه سمت سرور جهت ارتباط با برنامه اصلی
- ساخت تصویر سفارشی از لینوکس به همراه برنامه ی توسعه داده شده در QT
- اجرای تصویر سفارشی لینوکس روی بورد رزبری پای

پروژه ۲

عنوان: عملیات fuzzing روی UEFI با استفاده از پلتفرم Simics	**نوع پروژه:** کاربردی
ایمیل طراح: mahdi.bahreiny96@sharif.edu	

توضیحات

Fuzzing یک تکنیک تست نرم افزار خودکار است که تلاش می‌کند آسیب پذیری ها را با استفاده از ورودی‌های تصادفی پیدا کند. هنگامی که کاربر درخواست ورودی غیر از راه های مشخص شده می‌دهد، نرم افزار اغلب رفتار غیرقابل پیش‌بینی از خود نشان می‌دهد. عمل فازی وارد کردن مقادیر زیادی از ورودی‌های غیرمنتظره و ثبت نتیجه‌ی چیزی است که اتفاق می‌افتد. ایده این است که کاربر می‌تواند نرم‌افزار را کنترل کند و تشخیص دهد که آیا آسیب‌پذیری یا باگ در آن وجود دارد یا خیر. از آنجایی که فرمور سیستم ها نیز یک سیستم نرم افزاری است، Fuzzing یک تکنیک محبوب جهت ارزیابی امنیتی آن است زیرا اجازه می دهد بدون دسترسی به کد منبع، آسیب پذیری های نرم افزار را پیدا کرد.

UEFI یا Unified Extensible Firmware Interface یک واسط نرم‌افزاری بین سیستم‌عامل، میان افزار (Firmware) و سخت‌افزار است که قبل از بوت سیستم به اجرا در می آید. UEFI کنترل سیستم را بعد از روشن کردن آن و قبل از شروع به کار سیستم عامل به دست می گیرد.

هدف این پروژه انجام عملیات Fuzzing روی UEFI با استفاده از پلتفرم مجازی Simics است.

ابزار مورد نیاز

- پلتفرم EDK2
- شبیه ساز Simics
- ابزار AFL

شرح پروژه

در این پروژه بایستی مقاله ی ارائه شده توسط شرکت اینتل پیاده سازی مجدد شود. در این مقاله پلتفرم هدف که فرمور را اجرا می کند و شامل سخت افزار مجازی مورد استفاده توسط آن می باشد با استفاده از Simics شبیه سازی می شود. از AFL که یک ابزار جهت انجام تست Fuzzing است و پلتفرم مجازی Simics استفاده شده است تا عملیات Fuzzing روی UEFI انجام شود. از دیدگاه AFL تمام پلتفرم Simics به عنوان باینری تحت تست درنظرگرفته می شود. عملیات fuzzing روی دو بخش SMI Handler و Hardware I/O انجام می شود.

مراحل انجام پروژه

- مطالعه ی مقاله و آشنایی با مفاهیم اولیه
- آشنایی با برنامه نویسی UEFI با استفاده از پلفترم EDK2
- نصب و راه اندازی برنامه نوشته شده با استفاده از پلتفرم مجازی Simics
- انجام عملیات Fuzzing با استفاده از AFL

پروژه ۳

عنوان: طراحی و پیاده‌سازی سامانه موقعیت‌یابی محلی	**نوع پروژه:** کاربردی
ایمیل طراح: mahdi.bahreiny96@sharif.edu	

توضیحات

توان بسیار پایین سیگنال ماهواره‌های موقعیت‌یابی (مانند GPS) در سطح زمین سبب می‌گردد تا در صورت وجود مانع و از بین رفتن شرایط line of sight ماهواره‌ها قابل شناسایی و ره‌گیری نباشند، به همین جهت است که طراحی یک سامانه موقعیت‌یابی محلی می‌تواند منجر به موقعیت‌یابی با دقت بالا در هر دو محیط بیرون و فضای بسته گردد. یکی از بهترین روش ها بهره‌گیری از فناوری Ultra Wide Band (UWB) است. در این روش هر یک از ماژول‌های Tag و Anchor شامل فرستنده - گیرنده‌های یکسان DW1000 هستند که از اختلاف زمانی بین پیام‌های رد و بدل شده، فاصله بین یکدیگر را با الگوریتم TWR اندازه‌گیری می‌کنند. با اعمال الگوریتم‌های موقعیت‌یابی بر روی فواصل اندازه‌گیری شده Tag با Anchorهای تعبیه شده در نقاط مختلف محیط، موقعیت سه‌بعدی Tag در دستگاه مختصات محلی محاسبه می‌شود. در این پروژه با استفاده از موتور بازی سازی Unity شبیه سازی عملیات موقعیت یابی انجام می شود.

ابزار مورد نیاز

- Unity

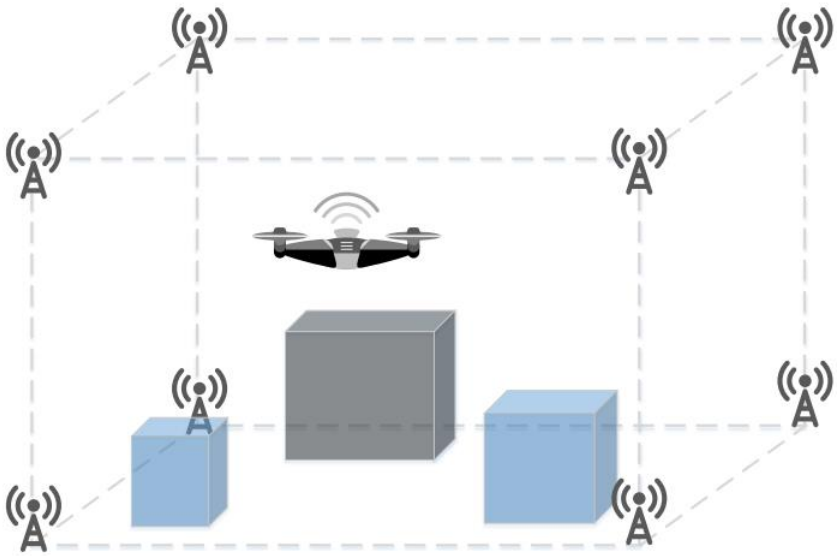
شرح پروژه

هدف از این پروژه شبیه سازی عملیات موقعیت یابی سه بعدی چندین هدف با استفاده از ماژول UWB است. در ابتدا بایستی در مورد این ماژول ها مطالعه شود و نحوه ی کار آن ها استخراج شود. در ادامه باید دو برنامه ی Unity توسعه داده شود. همان‌طور که بیان شد قرار است تا هر یک از Anchorها به صورت متناوب فاصله خود را با هر یک از Tagها اندازه‌گیری کرده و با حل معادلات، موقعیت سه‌بعدی Tag محاسبه شود.

در برنامه نخست بایستی عملکرد Anchor و Tag شبیه سازی شود. براساس نحوه ی کار ماژول های UWB بایستی جایگذاری Anchor ها در محیط سه بعدی مکعبی انجام شده و فاصله Anchor و Tagها برای برنامه ی دوم ارسال شود. توجه شود که در این برنامه تشخیص تعداد لازم Anchor برای موقعیت یابی دقیق Tagها ضروری است. توجه شود برای حل معادلات موقعیت‌یابی سه‌بعدی Tag به اطلاعات حداقل سه Anchor نیاز است، اما دقت لازم شاید تأمین نشود. مسئله چینش Anchorها برای رسیدن به دقت بهینه با تعداد محدود Anchor، یک مسئله بهینه‌سازی است که به صورت تئوری قابل حل است اما پیشنهاد می‌شود که یک هندسه اولیه متشکل از حداقل 6 عدد Anchor که به طور مساوی در سطح زمین و در ارتفاع پخش شده‌اند استفاده شود.

در برنامه ی دوم بر اساس داده های به دست آمده از برنامه ی نخست، بایستی الگوریتم های موقعیت یابی روی آن اجرا شده تا موقعیت هر Tag به دست آید. براساس داده های به دست آمده موقعیت مکانی Tag ها در محیط سه بعدی بایستی نمایش داده شود.

در این برنامه ها از حداقل 3 عدد Tag بایستی استفاده شود.



نمونه‌ای از چینش Anchorها در سامانه موقعیت‌یابی محلی

مراحل انجام پروژه

- آشنایی با نحوه ی کار ماژول های UWB
- شبیه سازی Anchor و Tagها در محیط سه بعدی
- تشخیص موقعیت موقعیت tagها بر اساس اطلاعات دریافتی و نمایش در محیط سه بعدی

پروژه ۴	
عنوان: تهیه‌ی پروژه‌ی benchmark برای آزمون سیستم‌های نهفته (ارتباطات)	**نوع پروژه:** کاربردی
ایمیل طراح: ahmad.saleh@ce.sharif.edu	

توضیحات:

آزمون و گزارش عملکرد یک سیستم پس از انجام طراحی و ساخت آن، جزو مهم‌ترین مراحل توسعه‌ی آن سیستم است. برای این منظور عمدتاً با توجه به هدف سیستم، مراحلی برای آزمون آن در نظر گرفته می‌شود و در نهایت یک گزارش متناسب با عملکرد سیستم، ارائه می‌شود. به فرآیند آزمون و کد مربوط به آن benchmark گفته می‌شود. در مقالات علمی و کاربردهای صنعتی، عمدتا برای مقایسه‌ی عملکرد دو سیستم از benchmark ها استفاده می‌شود. در این صورت امکان گزارش بهبودِ صورت گرفته در ایده یا سیستم جدید فراهم می‌شود. با توجه به گستردگی استفاده از سیستم‌های نهفته، وجود benchmark برای این سیستم‌ها اهمیت زیادی دارد. برای این منظور چندین پروژه‌ی مختلف در این زمینه تهیه شده است که در مقالات و ایده‌های مختلف حوزه‌ی سیستم‌های نهفته به آنها ارجاعات زیادی داده شده است. برای مثال دو پروژه‌ی Mibench[1] و Embench[2] از این دسته هستند. البته نمونه‌های دیگری همچون EEMBC[3] نیز وجود دارد که تمام اجزای آن تحت پروژه‌ای متن‌باز نیستند و برای دسترسی به برخی از عملکردها، نیاز به پرداخت هزینه است. طبیعتا استفاده از آنها در پروژه‌های پژوهشی توجیه خاصی ندارد. چهار اشکال اساسی متوجه پروژه‌های benchmark های حال حاضر هستند.

۱. قدیمی بودن برخی پروژه‌ها که باعث عدم تطابق بین عملکرد اصلی سیستم‌های نهفته‌ی امروزه و عملکردهایی که این پروژه‌ها آزمون می‌کنند، می‌شود.
۲. عدم کارا بودن برخی از عملکردهای مورد آزمون به صورت عمومی
۳. عدم ارائه‌ی شواهد آماری لازم برای میزان استفاده از یک عملکرد خاص مورد آزمون. (به عنوان مثال اگر الگوریتم ضرب دو ماتریس مورد آزمون قرار گرفته است، این عملکرد در چند درصد سیستم‌های نهفته مورد استفاده قرار می‌گیرد.)
۴. هزینه‌بردار بودن استفاده از برخی از این benchmark ها

به توجه به این اشکالات یک خلاء جدی در زمینه‌ی benchmark سیستم‌های نهفته در پژوهش و صنعت حس می‌شود. ما در پروژه‌ای نسبتا مفصل قصد داریم تا این خلاء را پوشش دهیم و با تقسیم این پروژه به بخش‌های مختلف و تخصیص آن به گروه‌های مجزا، به مرور زمان آن را تکمیل کنیم. از آنجایی که ما قصد ارائه‌ی شواهد لازم برای میزان کاربرد یک عملکرد خاص در سیستم‌های نهفته را داریم، بخش عمده‌ی انجام این پروژه‌ها به مطالعه و پژوهش برای ارائه‌ی یک آمار کاربردی نیاز دارد.

شرح پروژه:

در پروژه‌ی اول، ما قصد داریم تا بخش‌های مرتبط با عملکردهای مربوط به ارتباطات سیستم‌های نهفته را مورد بررسی قرار دهیم. به عنوان مثال i2c، SPI و UART جزو پروتکل‌های ارتباطی رایج هستند. اما پروتکل‌های ناشناخته‌تر دیگری نیز در حوزه‌ی سیستم‌های نهفته وجود دارد. در این پروژه باید ۵ پروتکل ارتباطی که مورد استفاده‌ترین پروتکل‌های ارتباطی در میان سیستم‌های نهفته هستند، تشخیص داده شود. برای این منظور لازم است با انجام پژوهش و ارائه‌ی لازم، این فاز شناسایی به درستی انجام شود. سپس در فاز بعدی لازم است تا کد

[1] https://github.com/embecosm/mibench
[2] https://github.com/embench/embench-iot
[3] https://www.eembc.org/products/

benchmark مربوط برای این ۵ پروتکل نوشته شود و برروی یک پلتفرم مشخص مورد آزمون قرار گیرد. نتایج این آزمون‌ها باید بر اساس معیار Worst Case Execution Time (WCET) بر اساس تعداد کلاک سیستم گزارش شود.

ابزار مورد نیاز

- بورد STM32

مراحل انجام پروژه

- آشنایی با پروتکل‌های ارتباطی رایج
- شناسایی پنج پروتکل مورد استفاده در سیستم‌های نهفته که بیشترین کاربرد را از نظر کمیت استفاده دارند
- پیاده‌سازی یا پیدا کردن یک کد آزمون برای این پروتکل‌های ارتباطی
- آزمون این کدها بر بستر یک پلتفرم مشخص

پروژه ۵	
عنوان: تهیه‌ی پروژه‌ی benchmark برای آزمون سیستم‌های نهفته (امنیت و رمزنگاری)	**نوع پروژه:** کاربردی
ایمیل طراح: ahmad.saleh@ce.sharif.edu	

توضیحات:

لطفا توضیحات پروژه‌ی شماره ۴ را مطالعه کنید.

شرح پروژه

امروزه یکی از کاربردهای بسیار متداول در سیستم‌های نهفته، انجام عملیات‌های امنیتی[۴] است. این عملیات‌ها عمدتا شامل امن کردن کانال ارتباطی و صحت‌سنجی داده‌های حساس می‌شود. بالتبع برای انجام این عملیات‌ها، از الگوریتم‌های استاندارد رمزنگاری استفاده می‌شود. مشابه پروژه‌ی ۱ در این پروژه ما قصد داریم تا الگوریتم‌های متداول رمزنگاری و پروتکل‌های امنیتی مهم این حوزه را شناسایی کنیم و آزمون‌های عملکردی متناسب با آنها را تهیه کنیم. لازم است گزارش پایانی benchmark بر اساس WCET و تعداد کلاک سنجیده شود.

ابزار مورد نیاز

- بورد STM32

مراحل انجام پروژه

- آشنایی با الگوریتم‌های رمزنگاری و پروتکل‌های امنیتی
- شناسایی الگوریتم‌ها و پروتکل‌های مورد استفاده در سیستم‌های نهفته که بیشترین کاربرد را از نظر کمیت استفاده دارند
- پیاده‌سازی یا پیدا کردن یک کد آزمون
- آزمون این کدها بر بستر یک پلتفرم مشخص

[4] secutity

پروژه ۶	
عنوان: تهیه‌ی پروژه‌ی benchmark برای آزمون سیستم‌های نهفته (یادگیری ماشین و یادگیری عمیق)	**نوع پروژه:** کاربردی
ایمیل طراح: ahmad.saleh@ce.sharif.edu	

توضیحات:

لطفا توضیحات پروژه شماره ۴ را مطالعه کنید.

شرح پروژه:

به تازگی انجام عملیات‌های یادگیری ماشین و یادگیری عمیق روی سیستم‌های نهفته، کاربردهای زیادی پیدا کرده است. پلتفرم‌های مختلفی برای این حوزه مانند TinyML نیز معرفی شده است. ما در این پروژه قصد داریم تا با شناسایی این کاربردها و روش استفاده از آن‌ها اقدام به طراحی benchmark برای حوزه‌ی یادگیری ماشین و یادگیری عمیق در سیستم‌های نهفته کنیم. پیشنهاد ما همچنان استفاده از بوردهای STM32 برای انجام benchmarking است زیرا این میکروکنترلر دارای کاربردهای بسیار زیادی در زمینه‌ی سیستم‌های نهفته است. اما اگر بر اساس دلایل و شواهد محدودیتی برای اجرا روی این پلتفرم وجود داشته باشد، می‌توانید در پروپوزال خود به این موضوع اشاره کنید. گزارش نهایی انجام benchmarking شما باید WCET بر اساس تعداد کلاک سیستم است.

ابزار مورد نیاز

- بورد STM32

مراحل انجام پروژه

- آشنایی با الگوریتم‌ها و پلتفرم‌های مختلف یادگیری ماشین و یادگیری عمیق سیستم‌های نهفته
- شناسایی الگوریتم‌ها و کاربردهای مورد استفاده در سیستم‌های نهفته که بیشترین نمود را از نظر کمیت استفاده دارند
- پیاده‌سازی یا پیدا کردن یک کد آزمون
- آزمون این کدها بر بستر یک پلتفرم مشخص

پروژه ۷

عنوان: پیاده سازی روش CLB در یک نرم‌افزار نهفته	**نوع پروژه:** کاربردی
ایمیل طراح: mghassab.uni@gmail.com	

در این پروژه، روش Cryptographically Obfuscated Logic Bomb یا به اختصار CLB که یکی از تکنیک‌های Anti repackaging می‌باشد باید روی یک نرم‌افزار نهفته در ثابت‌افزار BMC پیاده‌سازی و تست گردد.

ابزار مورد نیاز:

برد توسعه BMC

شرح پروژه:

در repackaging، فرد مهاجم به (فایل باینری) ثابت‌افزار اصلی یکسری دستور اضافه کرده و یک ثابت‌افزار جدید و مشتق شده از ثابت‌افزار اصلی می‌سازد. متأسفانه اکثر ثابت‌افزارها مکانیزم‌های تأیید یکپارچگی مناسب نداشته و در معرض حملات repackaging قرار دارند. علاوه بر این، بسیاری از راه‌حل‌های موجود برای دستگاه‌های اینترنت اشیا تنها بر روی بخشی از فرآیند بروزرسانی ثابت‌افزار تمرکز دارند (به عنوان مثال، از سرور به‌روزرسانی به دستگاه) و تأیید کاملی از ثابت افزار دانلود شده انجام نمی‌دهند و از این رو نمی‌توانند از یکپارچگی آن اطمینان حاصل کنند. البته برای این منظور راه حل‌هایی ارائه شده اما اکثر آن‌ها به یک عامل خارجی مثل کلیدهای امضا یا فناوری های ذخیره سازی ایمن نیاز دارند که می تواند استفاده از آنها را در پلتفرم‌هایی با منابع محدود، دچار مشکل کند.

بطور خلاصه در این ایده، در زمان ساخت محتویات logic bomb را رمزنگاری می‌کنند که این محتویات رمزنگاری شده در زمان اجرا یا ران تایم، باید رمزگشایی و اجرا شود. برای رمزنگاری محتویات logic bomb، لازمه ما کلید را در فایل اجرایی قرار بدیم، اما با این کار فرد مهاجم خیلی راحت و با یک بررسی ساده می‌تواند کلید را با تکنیک‌های مهندسی معکوس پیدا کند. برا ی جلوگیری از سرقت کلید، بهترین راه این است که ما از منطق اجرای خود ثابت‌افزار برای مخفی کردن کلید استفاده کنیم. می‌دانیم یک متغیر، مقادیر مختلفی در طول اجرا خواهد داشت، بهترین کار برای مخفی کردن کلید، این است که یکی از همین مقادیری که متغیر در مدت زمان اجرا دارد را به عنوان کلید قرار بدیم، با این کار دیگر نیازی نیست بصورت مستقیم به مقدار کلید در کد اشاره کنیم.

در این پروژه ابتدا باید این مقاله مطالعه و بررسی شود تا با مباحث نظری و نحوه کار تکنیک CLB آشنایی صورت گیرد، سپس باید ابزار معرفی شده در این مقاله را روی یک نرم‌افزار نهفته در ثابت‌افزار BMC مانند bmcweb اعمال کرده و تغییرات مورد نیاز جهت تطبیق‌پذیری ابزرا با آن اعمال شود. در نهایت باید کارایی روش CLB بررسی شود.

پروژه ۸

عنوان: طراحی سامانه تحمل‌پذیر اشکال پایش دما	**نوع پروژه:** کاربردی
ایمیل طراح: amir.r.moezi@gmail.com	

توضیحات

امروزه سامانه‌های الکترونیکی کاربرد زیادی در زندگی روزمره ما پیدا کرده‌اند. در برخی کاربردها عملکرد درست آن‌ها به یک ضرورت تبدیل شده است و در صورت بروز مشکل برای آن‌ها خساراتی به وجود می‌آید. برای مثال، یک سیستم کنترل دمای یک محیط صنعتی را در نظر بگیرید که وظیفه ثابت نگه داشتن دمای محیط را بر عهده دارد. اگر به هر دلیلی این سیستم دچار اختلال شود، کنترل دمای آن محیط با مشکل مواجه می‌شود که می‌تواند بسیار خطرناک باشد. به طور کلی، برای اطمینان از عملکرد درست یک سیستم، مجموعه‌ای از تکنیک‌ها به کار برده می‌شود تا سامانه مدنظر اصطلاحا "تحمل پذیر اشکال" شود تا در صورت بروز اشکال در یک قسمت، سامانه همچنان به کار خود ادامه دهد.

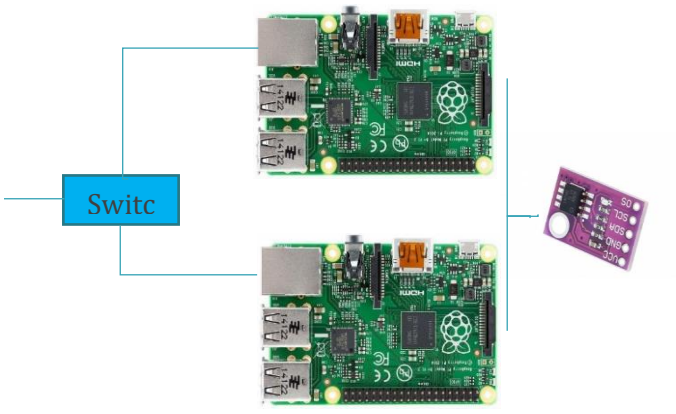
یکی از تکنیک‌های تحمل‌پذیری اشکال، بکارگیری سامانه‌های یدکی است. این به این معنا است که یک سیستم مشابه سیستم اصلی وجود دارد که در صورت بروز مشکل برای سیستم اصلی وارد عمل شده و کار را ادامه دهد. سه نوع سامانه یدکی وجود دارد: سامانه یدکی سرد، گرم و داغ. سامانه یدکی سرد کاملا خاموش است و پس از بروز مشکل برای سامانه اصلی روشن شده و کار را ادامه می‌دهد. سامانه گرم روشن است ولی عملیاتی نیست و پس از بروز مشکل برای سامانه اصلی عملیاتی شده و کار را ادامه می‌دهد. سامانه داغ همزمان با سامانه اصلی عملیاتی و در حال کار است بلافاصله پس از بروز مشکل برای سامانه اصلی کار را برعهده می‌گیرد.

شرح پروژه

در این پروژه، هدف طراحی یک سامانه پایش دمای تحمل پذیر اشکال است. دو برد رزبری‌پای وجود دارد که یکی سامانه اصلی و دیگری سامانه یدک است. یک سنسور دما نیز وجود دارد که سامانه اصلی دما را از آن می‌خواند و گزارش می‌کند. دانشجو باید مکانیزمی را پیاده‌سازی کند که سامانه یدک پس از بروز مشکل برای سامانه اصلی متوجه شود و کار را بر عهده بگیرد. در این پروژه باید هر دو سامانه گرم و داغ پیاده‌سازی شوند. استفاده از نرم‌افزار یا سرویس خاص و یا بکارگیری قطعه جدید به صلاحدید دانشجو امکان پذیر است. در سامانه یدک گرم، تاخیر راه‌اندازی سامانه یدک باید حداقل باشد و مقدار آن گزارش شود. از سوی دیگر، در سامانه یدک داغ اطلاعات سنسور دما در صورت بروز مشکل برای سامانه اصلی، بدون وقفه باید به کاربر ارسال شود.

توجه:

- از سیستم‌عامل‌های مختلفی می‌توان برای رزبری‌پای استفاده کرد. در این پروژه، استفاده از سیستم‌عامل OpenBMC بر روی بردهای رزبری‌پای، نمره امتیازی قابل توجهی دارد.
- امکان تغییر در معماری پیشنهادی زیر، افزودن یا کاهش قطعات یا استفاده از یک نرم‌افزار خاص با تایید مسئول پروژه امکان‌پذیر است.



پروژه ۹

عنوان: شبیه‌سازی سامانه کنترل دمای سرور با رزبری‌پای	**نوع پروژه:** کاربردی
ایمیل طراح: amir.r.moezi@gmail.com	

توضیحات

امروزه سرورها بخش جدایی‌ناپذیر شبکه اینترنت هستند. بنابراین، کارکرد درست آن‌ها الزامی است. شرایط محیطی سرورها باید بگونه‌ای باشد تا خللی در عملکرد آن ایجاد نکند. از جمله شرایط محیطی ضروری، دمای مناسب داخل سرور است. کنترل دمای داخل سرور توسط یک تراشه مستقل از CPU به نام BMC انجام می‌شود. تراشه BMC با استفاده از سنسورهای دمای متصل به خود، دمای بخش‌های مختلف سرور را اندازه‌گیری کرده و بر اساس آن‌ها با تنظیم سرعت فن، دمای داخل سرور را کنترل می‌کند.

شرح پروژه

در این پروژه قصد داریم تا این مدل کنترل دما را پیاده سازی کنیم. به این منظور، از برد رزبری‌پای به عنوان تراشه BMC استفاده می‌شود. تراشه BMC برای کار نیاز به یک سیستم‌عامل تعبیه شده شده دارد. معروف‌ترین سیستم‌عامل متن‌باز موجود برای تراشه‌های BMC، سیستم‌عامل OpenBMC است که امکان استفاده آن بر روی بردهای رزبری‌پای نیز وجود دارد که در این پروژه از این سیستم‌عامل استفاده خواهد شد. با اتصال سنسور دما به همراه فن به برد رزبری‌پای شبیه‌سازی سامانه کنترل دمای سرور با رزبری‌پای انجام خواهد شد.

مراحل انجام پروژه به این شرح است:

۱) کامپایل OpenBMC برای برد رزبری‌پای و اجرای آن بر روی برد

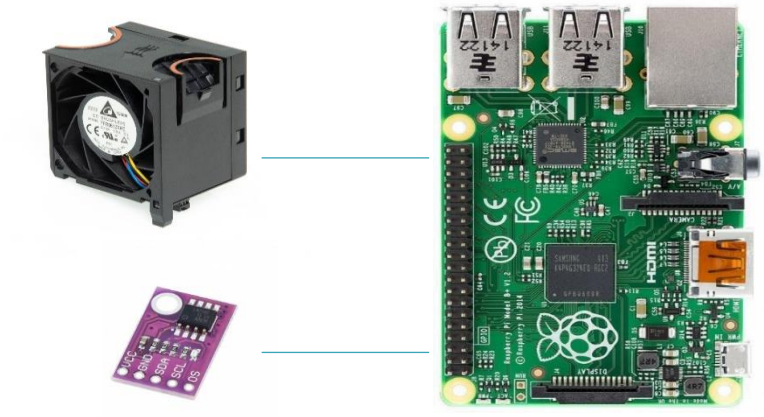
۲) راه‌اندازی یک فن متصل به برد رزبری‌پای

۳) خواندن دما از یک سنسور دمای متصل به برد رزبری‌پای

۴) کنترل سرعت فن با توجه دمای سنسور (به صورت state machine).

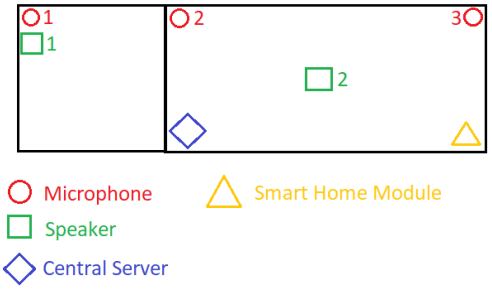
۵) کنترل سرعت فن با توجه دمای سنسور (با یک حلقه کنترلی PID).

۶) (امتیازی) استفاده از رابط کاربری تحت وب OpenBMC برای نمایش گرافیکی دما و سرعن فن



پروژه ی ۱۰	
عنوان: دستیار صوتی مبتنی بر ChatGPT در خانه هوشمند	**نوع پروژه:** کاربردی
ایمیل طراح: mahdi.barati765@sharif.edu	

توضیحات: دستیار صوتی همان‌طور که از اسمش هم پیداست، از طریق فرامین صوتی کار می‌کند. شما می‌توانید دستورهای صوتی (حتی دستورات پیچیده) را به دستگاه فرمان داده تا این ابزارهای دیجیتالی، دستورهای شما را در سریع‌ترین حالت ممکن اجرا کنند از طرفی بهره‌برداری از دستیار صوتی برای نابینایان، سالمندان و کودکان یک نیاز ضروری است. یکی از کاربری های مورد استفاده از فرمان های صوتی در سیستم خانه هوشمند است. همان طور که در شکل زیر مشخصه برای پوشش زیرساخت سخت افزاری از دستگاه های بیسیم مبتنی بر شبکه WiFi برای میکروفن و اسپیکر جهت پوشش دو اتاق این خانه هوشمند در نظر گرفته شده است. این خانه هوشمند شامل یک سرور مرکزی جهت برقرار ارتباط با اینترنت و یا کاربری جهت پردازش سنگین است. هر ماژول بیسیم میکروفن و یا اسپیکر دارای هسته میکروکنترلر ESP32 می باشد. نوع میکروفن INMP441 و نوع ماژول راه‌انداز اسپیکر UDA1334A که خروجی AUX جهت اتصال به سیستم صوتی خانه هوشمند فراهم می‌کند. در واقع 3 ماژول میکروفن و دو ماژول اسپیکر و یک ماژول ESP32 کنترلر خانه هوشمند که یک سنسور دما و سه کانال خروجی روشنایی به ان متصل است.



سناریو کاربری استفاده از این زیرساخت جهت تحلیل فرمان صوتی مبتنی بر API ChatGPT است. لازم به ذکر است فرمان های صوتی به صورت فارسی در این پروژه در نظر گرفته شود. در این سناریو هر کدام از ماژول های میکروفن که سریع تر کلمه Wake up را تشخیص داد، در ادامه Wake up word ، مکالمه کاربر را به صورت فایل WAV در حافظه داخلی ESP32 ذخیره کند و سپس فایل ذخیره شده برای سرور مرکزی خانه هوشمند ارسال گردد یا به صورت stream هر ماژول میکروفن بعد تشخیص کلمه Wake up ، صدا را برای سرور مرکزی خانه هوشمند ارسال کند. مدت زمان timeout پس از تشخیص کلمه Wake-Up جهت ارسال یا ذخیره مکالمه ماکزیمم ۳۰ ثانیه است. استفاده از Wake up word به واسطه مشکلات حریم خصوصی کاربر درنظر گرفته شده است و صرفا در بعد از گفتن این کلمه ذخیره یا ارسال صدا انجام می شود. لازم به ذکر است جهت تشخیص کلمه wake up با استفاده از مدل های یادگیری ماشین به صورت محلی در پردازنده ESP32 در ماژول میکروفن انجام می پذیرد. در این سناریو اگر فرمان صوتی از میکروفن یک اتاق به خصوص باشد می بایست پاسخ فرمان صوتی از طریق ماژول اسپیکر همان اتاق داده شود. در این سناریو بدلیل امنیت صرفا دستگاه سرور مرکزی دسترسی مستقیم به اینترنت دارد. در این سناریو کاربر درخواست های مختلف می تواند داشته باشد. بعضی از این دستورات به طور مشخص از پیش تعریف شده هستند در سناریو خانه هوشمند، به عنوان مثال کاربر میخواهد از دما داخل اتاق اطلاع پیدا کند و یا کانال های روشنایی خاصی را روشن یا خاموش کند. که این فرمان اجرایی از طریق ماژول خانه هوشمند انجام می پذیرد. از طرفی کاربر سوالات عمومی از ChatGPT دارد که می بایست فرمان وی به صورت صوتی پاسخ داده شود. از طرفی کاربر مدنظر دارد بتواند داده های history در خانه هوشمند را از دیتابیس بخواند. بدین منظور به واسطه داده سنسور دما ماژول کنترلر خانه هوشمند دما خانه را هر ۵ دقیقه یکبار به همراه TimeDate مشخص در دیتابیس محلی سرور خانه هوشمند ذخیره می شود. در این سناریو کاربر فرمان صوتی با TimeDate مشخصی اعلام می‌کند و مدنظر دارد دما ذخیره شده در ان زمان معلوم اطلاع پیدا کند. در این سناریو از یک دیتابیس استاندارد استفاده گردد. همچنین داکیومنت و سورس کد هایی از پروژه مرتبط تیم های پیشین در اختیار شما قرار می‌گیرد.

پروژه ی ۱۱	
عنوان: پیاده سازی شبیه سازی مجدد مقاله با عنوان Optimizing Tradeoff Between Learning Speed and Cost for Federated-Learning-Enabled Industrial IoT	**نوع پروژه:** کاربردی
ایمیل طراح: mahdi.barati765@sharif.edu	

توضیحات: این پروژه با هدف پیاده‌سازی مجدد و ارزیابی عملیاتی مقاله عنوان صورت می‌گیرد که به بررسی ترکیب اینترنت اشیاء صنعتی (IIoT) و یادگیری مشارکتی (Federated Learning) به‌عنوان راهکاری برای بهبود عملکرد سیستم‌های صنعت نسل ۴ پرداخته است. هدف اصلی این پروژه، بازپیاده‌سازی الگوریتم پیشنهادی مقاله جهت بهینه‌سازی همزمان ارتباطات با در محیط لبه شبکه، تخصیص منابع، ظرفیت محاسباتی و توان ارسال دستگاه‌های IIoT در یک سیستم FL سه‌لایه است. این الگوریتم با تمرکز بر موازنه میان سرعت یادگیری و هزینه انرژی طراحی شده است و به حل یک مسئله بهینه‌سازی غیرمحدب و پیچیده با استفاده از تکنیک‌های تجزیه مسئله و بهینه‌سازی متناوب می‌پردازد. تیم اجرایی در این پروژه، الگوریتم‌های موردنظر را با بهره‌گیری از داده‌ها و پارامترهای واقعی شبیه‌سازی کرده و به ارزیابی عملکرد سیستم از نظر بهبود کاربرد یادگیری و دستیابی به موازنه بهینه میان سرعت یادگیری و هزینه انرژی خواهد پرداخت. نتایج حاصل از این پیاده‌سازی مجدد، امکان مقایسه دقیق عملکرد روش‌های مختلف و تایید یا نقد نتایج ارائه‌شده در مقاله اصلی را فراهم خواهد کرد.

لینک دسترسی به مقاله در IEEE

Optimizing Tradeoff Between Learning Speed and Cost for Federated-Learning-Enabled Industrial IoT | IEEE Journals & Magazine | IEEE Xplore

پروژه ی ۱۲	
عنوان: پیاده سازی عملی اجرای یادگیری مشارکتی در سخت افزار دارای GPU/CPU	**نوع پروژه:** کاربردی
ایمیل طراح: mahdi.barati765@sharif.edu	

توضیحات: ابزار شبیه سازی HFL-Energy یک ابزار برای تحلیل انرژی مصرفی دستگاه های مشارکت کننده در فرایند یادگیری مشارکتی (Federated Learning) است. این ابزار شبیه سازی انرژی مصرفی و زمان آموزش مدل را به صورت شبیه سازی براساس قدرت پردازشی، تعداد سمپل های داده های موجود و ... برای هر دستگاه محاسبه می‌کند. در این پروژه مدنظر هست آموزش مدل دستگاه کاربران از محیط شبیه سازی فعلی بر روی دستگاه واقعی برون سپاری گردد و زمان آموزش واقعی اندازه گیری گردد. اقداماتی از تیم های پیشین جهت ارسال مدل های خام بر روی دستگاه از طریق برنامه نویسی سوکت انجام شده است. در این پروژه دستگاه های واقعی در انواع NVIDIA Jetson Nano, NVIDIA Jetson TX2 NX و رزبری پای های ورژن ۳ و ۴ به همراه Orange Pi PC Plus و Orange pi 5 موجود است که برنامه Client بر روی این دستگاه ها پیاده سازی می‌گردد. در این پیاده سازی به طراحی یک رابط جهت تنظیم فرکانس کاری GPU/CPU هر دستگاه به صورت واقعی انجام می پذیرد. همچنین با استفاده از تکنیک پارتیشن سازی داده داده های هر دستگاه بین CPU و GPU تقسیم می‌گردد. لازم به ذکر است خروجی این پیاده سازی، سازگاری کامل با کارکرد ابزار شبیه ساز را داشته باشد، بدین صورت که در ابزار شبیه ساز بتوان دستگاه ها را چه به صورت شبیه ساز یا به صورت واقعی تعریف و تنظیمات هر کدام را مشخص کرد و در کنار یک دیگر در فرایند یادگیری مشارکتی فعال باشند. اجرای فرایند یادگیری میتواند به صورت سریال دستگاه به دستگاه انجام شود و اجرای موازی نیاز نیست. همچنین سورس کد و اقداماتی از تیم های پیشین جهت پیشبرد این پروژه موجود است که در اختیار تیم جدید قرار می‌گیرد. همچنین بعضی از محدودیت ها دستگاه ها مانند اینکه صرفا CPU دارا باشند و قابلیت کاهش فرکانس کاری (DVFS) را نداشته نباشند و صرفا تک فرکانس عمل کنند، وجود دارد.

لینک ابزار شبیه سازی:

mbta009/HFL-Energy: Simulation of Hierarchical Federated Learning for Analyzing Client Energy Consumption (github.com)

پروژه ی ۱۳	
عنوان: طراحی gateway صنعتی مبتنی بر OpenWRT جهت پشتیبانی از پروتکل ها و ارتباطات صنعتی	**نوع پروژه:** کاربردی
ایمیل طراح: mahdi.barati765@sharif.edu	

توضیحات: هدف این پروژه، طراحی و پیاده‌سازی یک Gateway صنعتی مبتنی بر سیستم‌عامل OpenWRT است که به منظور پشتیبانی از پروتکل‌های صنعتی Profinet ، Profibus-DP و EtherCAT طراحی شده و قابلیت تبدیل آن‌ها به پروتکل‌های پرکاربرد اینترنت اشیا نظیر OPC UA و MQTT را نیز داراست. انتخاب OpenWRT به دلیل انعطاف‌پذیری بالا، قابلیت سفارشی‌سازی گسترده و پشتیبانی از طیف وسیعی از پلتفرم‌های سخت‌افزاری صورت گرفته است. این سیستم‌عامل امکان تطبیق و استفاده از سخت‌افزارهای مختلف را برای توسعه‌دهندگان فراهم می‌کند و در نتیجه گزینه‌ای ایده‌آل برای طراحی Gateway های صنعتی به شمار می‌رود. در این پروژه، برای پیاده‌سازی Gateway صنعتی از پلتفرم‌های سخت‌افزاری مانند Raspberry Pi یا دیگر سیستم‌های مبتنی بر لینوکس که با OpenWRT سازگاری دارند، استفاده خواهد شد. این انتخاب به دلیل سازگاری بالا با OpenWRT و همچنین دسترسی به منابع پردازشی مناسب و ماژول‌های توسعه‌پذیر انجام شده است. این پلتفرم‌ها با هزینه پایین و دسترسی آسان به منابع سخت‌افزاری، امکان توسعه و تست نرم‌افزارهای مرتبط با Gateway صنعتی را برای توسعه‌دهندگان فراهم می‌کنند. یکی از اهداف اصلی این پروژه، بهبود قابلیت تحمل‌پذیری اشکال (Fault Tolerance) در فریمور سفارشی‌سازی‌شده OpenWRT است. برای دستیابی به این هدف، از پورت JTAG به عنوان ابزاری برای تزریق خطا در سطوح مختلف فریمور و سخت‌افزار استفاده خواهد شد تا رفتار و پایداری Gateway در مواجهه با اشکالات احتمالی مورد بررسی دقیق قرار گیرد. علاوه بر این، آزمایش‌های تزریق خطا (Fault Injection) در لایه‌های مختلف فریمور و پروتکل‌های ارتباطی به منظور ارزیابی تحمل‌پذیری سیستم و شناسایی نقاط ضعف آن انجام خواهد شد. این آزمایش‌ها به تیم توسعه کمک می‌کند تا با تحلیل دقیق‌تر نقاط آسیب‌پذیر، عملکرد و قابلیت اطمینان سیستم را بهبود بخشیده و در نهایت به افزایش پایداری Gateway صنعتی منجر شود. لازم به ذکر است جهت تست، دستگاه های متصل به gateway میتوانند ابزار های شبیه ساز پروتکل یا یک دستگاه صنعتی واقعی مانند PLC باشد.