VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER SYSTEMS

SYSTÉM PRO ZABEZPEČENÍ A STŘEŽENÍ AUTOMOBILU

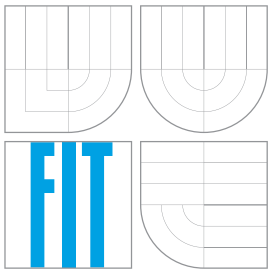
BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE
AUTHOR

VÍT ŠIROKÝ

BRNO 2008

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY



FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
ÚSTAV POČÍTAČOVÝCH SYSTÉMŮ

FACULTY OF INFORMATION TECHNOLOGY
DEPARTMENT OF COMPUTER SYSTEMS

SYSTÉM PRO ZABEZPEČENÍ A STŘEŽENÍ AUTOMOBILU

SYSTEM FOR GUARDING AND SECURING AUTOMOBILES

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — VÍT ŠIROKÝ
AUTHOR

VEDOUCÍ PRÁCE — Ing. JOSEF STRNADEL, Ph.D.
SUPERVISOR

BRNO 2008

Abstrakt

Tato práce se zabývá problematikou zabezpečení osobních automobilů. Začíná se obecným popisem vykradení a krádeží automobilů. Dále jsou diskutovány různé možnosti zabezpečení. Následuje popis návrhu systému a realizace jeho části s ohledem na nízký příkon. Speciální část textu se věnuje popisu možností realizace v mém starém autě Škoda Favorit. Závěr se zabývá možnnostmi rozšíření.

Klíčová slova

Zabezpečení, automobil, systém, návrh, FITkit

Abstract

This thesis discusses problem of guarding and securing automobiles. It starts by description of autocrime, continues by discussing about different ways of securing cars. Next, there is concept of my system and description of system realization with low power consumption. Special part of this chapter discusses ways of realization in my old Škoda Favorit car. Last chapter discusses some possibilities of extensions.

Keywords

Security, car, system, design, FITkit

Citace

Vít Široký: Systém pro zabezpečení a střežení automobilu, bakalářská práce, Brno, FIT VUT v Brně, 2008

Systém pro zabezpečení a střežení automobilu

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana Ing. Josefa Strnadela, Ph.D.

.....
Vít Široký
29. července 2008

Poděkování

Moje poděkování patří zejména vedoucímu této práce panu Ing. Josefu Strnadelovi, Ph.D. za jeho aktivní přístup k práci a za vypsání tohoto zajímavého tématu. Dále bych rád poděkoval panu Ing. Bohuslavu Křenovi, Ph.D., díky němuž jsem našel chuť k práci v LaTeXu, kterým jsem pohodlně a bez zbytečných komplikací vysázel tento dokument. A dále bych rád poděkoval svému tátovi panu Romanu Širokému, který mě toho hodně o autech naučil a poskytl mi spoustu zajímavých informací o autech starších i novějších.

Obsah

Kapitola 1

Úvod

V členských zemích Evropské unie je každoročně odcizeno přibližně 1,2 miliónu vozidel. Tato trestná činnost působí každoročně škody přesahující 15 mld. eur. V roce 2006 bylo odcizeno 20.996 motorových vozidel (20.175 dvoustopých vozidel a 821 jednostopých vozidel). Přes 90 % krádeží vozidel se týká dvoustopých motorových vozidel. Objasněnost se u této kategorie pohybuje od poloviny devadesátých let kolem 16 %. V roce 2006 dokonce pouhých 14 %. Znamená to, že v průměru bylo na území České republiky v roce 2006 odcizeno 57 motorových vozidel denně [17].

Vývoj osobních automobilů probíhá dosti vysokým tempem. Výrobci postupně zlepšují téměř všechny jejich parametry. Narůstají výkony motorů, výrazným způsobem stoupá pasivní i aktivní bezpečnost, cestování je stále pohodlnější, cestující i řidič mají k dispozici špičkovou techniku.

Příkladů lze uvést mnoho, jednodušší či složitější autorádia, navigační systémy, DVD přehrávače, telefony, televizní přijímače, parkovací asistenty a další. K tomu patří i jiné věci - kožené doplňky či celý interiér vyvedený v kvalitní kůži, elektricky ovládané veškeré příslušenství vozu, špičkové světlomety či kola z lehkých slitin.

Kromě komfortu posádky ovšem roste i riziko. Veškeré vybavení vozu má svou cenu, v případě originálního vybavení vozidla jeho výrobcem bývá cena velmi vysoká. To je ovšem velmi nebezpečné, jelikož ani moderní doba se nedovede vypořádat s lidmi, kteří se živí nepoctivě. A když pak takový zloděj vidí někde zaparkované auto s něčím cenným, co mu zabrání, aby si to prostě vzal, a někomu dalšímu na černém trhu pod cenou prodal? Co brání vzít si celý vůz? Jedině jeho zabezpečení.

Kapitola 2

Ohrožení automobilu

2.1 Odcizení

Zejména luxusní vozy tvoří velmi zajímavé zboží na černém trhu. Kradou se náhodně, ale také na objednávku.

Způsobů jak takové vozidlo odcizit je celá řada. Nejjednodušší auta lze za několik sekund otevřít díky minimálnímu zabezpečení zámku a nastartovat pomocí propojení vhodných vodičů tak, jak to lze vidět např. ve filmu. Soudobé vozy už jsou běžně vybavovány systémy, které to znesnadní, avšak vždy lze automobil prostě naložit a odvézt.

2.2 Vykradení

Kromě mnohdy velmi cenné výbavy vozu může jeho uživatel v autě ponechat i další předměty. Častý problém představuje zapomenutý mobilní telefon, PDA, nebo kufřík či notebook odložený na zadním sedadle. Takový cenný předmět může přilákat zloděje.

Základním předpokladem k vykradení vozidla je přístup do vozidla. Tomu samozřejmě bráníme uzamčením. Ovšem zámek je věc nedokonalá a lze překonat. Obvykle platí, že starší zámky jsou překonatelné snadno, pomocí šikovných rukou a několika nástrojů zámek zloděj odjistí a dveře jsou otevřeny. Proti tomu výrobci bojují stále lepší konstrukcí. Např. u starých vozidel Škoda 120 stačilo vytrhnout těsnící lištu u okna a vhodným předmětem (například i onou lištou) zatlačit správným způsobem na mechaniku zámku, který v důsledku toho odemkl. To už dnes samozřejmě nepřichází v úvahu.

2.3 Krádež vnějšího příslušenství

Na každém autě je venku něco, co lupiče zaujme. Např. disky mohou stát i několik desítek tisíc korun kus, xenonové světlomety jsou také velmi drahé. Proto se používá vhodných konstrukčních řešení tak, že montážní prvky jsou umístěny na místa z venkovního prostoru nepřístupná, případně se používají bezpečnostní šrouby s nestandardní hlavou apod.

Kapitola 3

Možná zabezpečení

Při návrhu zabezpečení je nutné zodpovědět si otázku, proti čemu vlastně chceme zabezpečit vozidlo? Logicky se nabízí snaha zabezpečit *proti všemu*. Ovšem ne všechno můžeme předpokládat a tak se snažíme vyjít ze známých skutečností. Celý systém tak musí uvažovat mnoho faktorů a být dosti rozsáhlý. Takový systém zabezpečení nelze vytvořit jinak, než dekompozicí na jednodušší části.

3.1 Zabezpečení proti vykradení

Zde nejde o nic jiného, než nepustit pachatele do auta. To je naprostý základ. S touhle problematikou musí počítat už konstruktéři auta při navrhování karoserie, zvláště pak dveří, jejich rámů a zámků. Například zámek ve dveřích, do něhož obvykle vkládáme klíč by neměl příliš přesahovat přes hranu dveří směrem do exteriéru vozu. Proč? Protože právě zámek je největší slabina. A přesah zámku umožní jeho uchycení nástrojem, například kleštěmi, a násilné otočení vedoucí k destrukci vnitřního mechanismu a odjištění dveří.

Nesmíme také umožnit otevření provázkem nebo drátem. K tomu stačilo pouze zatáhnout smyčku kolem kolíčku, který slouží k zamknutí nebo odemknutí dveří z interiéru vozu. Používal se v dobách, kdy nebylo v autech zvykem centrální zamykání. Některé automobily, např. Škoda Favorit, pak dovedly tuto slabinu k dokonalosti tím, že nepoužívají kolíček hladký, ale naopak designově krásně vyvedený kolík s hlavičkou podobnou hřibu. Naproti tomu třeba novější Felicie už byly vybavovány plastovým krytem okolo kolíčku klínovitého tvaru, který zajistil sesmeknutí smyčky.

Logicky vyplývá otázka, jak se tam dostaneme provázkem či drátem? Jednou možností je zapomnětlivý řidič, např. i kuřák (ti velmi často mají trochu otevřené okénko kvůli odklepávání popela), který zapomene zavřít okno. Stačí malá štěrbina. Dále není takový problém prostě přihnout horní roh dveří - opět se to týká starých měkkých konstrukcí - a pod těsněním prostrčit drát.

Určitě existují i další způsoby, jakými vniknout do auta. Ale protože toto není manuál pro zloděje, nemohu a ani nechci je zde rozvádět. Je věcí konstruktérů aut, aby vytvářeli stále nová vylepšení a znepříjemňovali tak vloupání. Ale ani nejlepší zámek na nejlepších dveřích nic ne;zmůže, když se použije metoda velmi jednoduchá, zato však velmi účinná - prostý úder do skla vozu. Další komentář k metodě myslím není třeba.

3.2 Zabezpečení proti odcizení celého auta

Když pomineme možnost odtahu, pak odcizení celého auta sestává ze tří základních bezprostředně navazujících kroků:

1. Vniknutí do vozu
2. Překonání mechanických zabezpečení
3. Nastartování vozu a odjezd

Z toho plyne, že kvalitní zabezpečení proti vykradení vozu pomůže významným způsobem se zabezpečením proti celému odcizení.

3.2.1 Mechanické zabezpečení

- Zcela běžnou věcí je uzamčení řízení po vytažení klíčku ze spínací skříňky. Lupič pak bez klíčku nemůže otáčet volantem a ovládání auta je pak nemožné až do okamžiku, kdy zámek zničí.
- Další věcí je vnější zámek na volantu. Jedná se zpravidla o tyč, s nějakým kloubem nebo posuvným zařízením. V odemknutém stavu lze s tyčí pohodlně manipulovat a po zaparkování vozidla ji nasadit na volant takovým způsobem, aby ji nebylo možné po uzamknutí sejmout. Tyč na volantu potom zabrání krouživému pohybu volantem, umožní jej pouze v minimálním rozsahu, a tak potom nelze vozidlo samozřejmě řídit. Podobné zařízení existuje i na pedály.
- Zámek řazení umožní uzamčení mechanismu řazení, obvykle v poloze, kdy je zařazen zpětný chod. S takto uzamčeným řazením je pochopitelně velmi těžké ovládat vozidlo a některé pojišťovny dokonce promítají zámek řadící páky do ceny pojistného. Systém má proti výše zmíněným zámkům volantu či pedálů jednu výhodu: je napevno nainstalován ve voze a je tak mnohem složitější jej překonat. Běžně dostupnou tyč na volant můžeme obejít například i tak, že prostě volant demontujeme a nasadíme jiný. Vypadá to absurdně, ale u krádeže na objednávku, kdy zloděj ví, pro jaké auto jde, není problém jiný volant přinést. Např. Česká pojišťovna poskytuje při pojištění vozidla slevu 10% z výše pojistného, je-li vozidlo vybaveno mechanickým zabezpečovacím zařízením, které musí blokovat převody vozidla [6].

3.2.2 Elektronika

Mějme hypotetickou situaci: Auto, do kterého se obtížně vnikne, a které má zároveň uzamčený volant, řízení i pedály. A zloděj stále trvá na tom, že jej chce ukrást. Pokud překoná mechanické zabezpečení, moc mu v tom nebrání. Zde přichází ke slovu elektronika.

Sice nemůžeme udělit pachateli eklektický šok, protože by mohlo dojít k poškození jeho zdraví, ale můžeme mu život znepříjemnit. V první řadě mu můžeme elektronické prvky v autě odstavit. A není to zrovna slabá zbraň. Elektrickou energii potřebuje pro svou funkci mnoho důležitých součástí vozidla:

- Startér
- Palivové čerpadlo

- Veškeré řídicí jednotky
- Automatická převodovka
- Zapalování(karburátorové motory)
- a další

Potíží takového zařízení může být ale hned několik: spotřeba elektrické energie a nalezení zařízení a jeho vyřazení z provozu. První problém se nemusí projevit, když jezdíme se svým autem každý den, kdy vybíjení baterie stálým příkonem elektronického systému stihneme bez problémů dobíjet jízdou (jak známo, auta mají alternátor, a potřebnou elektrickou energii si vyrábí a přitom dobíjí baterii, která tak slouží vlastně jen k nastartování a vykrývání případných kolísání napětí, které ovšem nejsou úplně běžnou záležitostí). Potíže ale objevíme v okamžiku, kdy odletíme na týden do zahraničí a necháme auto doma. Jak jinak než se zapnutým zabezpečením. A pak se vrátíme a auto nepůjde nastartovat, protože z baterie bylo odebráno příliš mnoho energie a její napětí již pokleslo. Proto je třeba volit systém s ohledem na nízký příkon.

Další problém vzniká, když systém zloděj najde a prostě ho zruší. Na autosalonu AMI Leipzig 2008 jsem si prohlédl celou řadu automobilů. Od běžných aut na nákup či pro rodinnou dovolenou, až po absolutní špičku, kterou představují německé speciality značek Porsche, BMW, Mercedes a Audi. Všechna tato auta jsou kompletně vybavena elektronikou, vše ovládají servomotory, například i takovou samozřejmost jakou je pro mě zavírání a otevírání víka zavazadlového prostoru. Nikdy by mě nenapadlo, že to nebudu dělat rukou, je to snadný pohyb. Pokud v takovém autě někam umístíme elektronický zabezpečovací systém, který bude odpojovat vhodné části, je nepravděpodobné, že jej pachatel najde a vyřadí z provozu, protože se nevyzná v tom, který vodič do svazku patří a který ne.

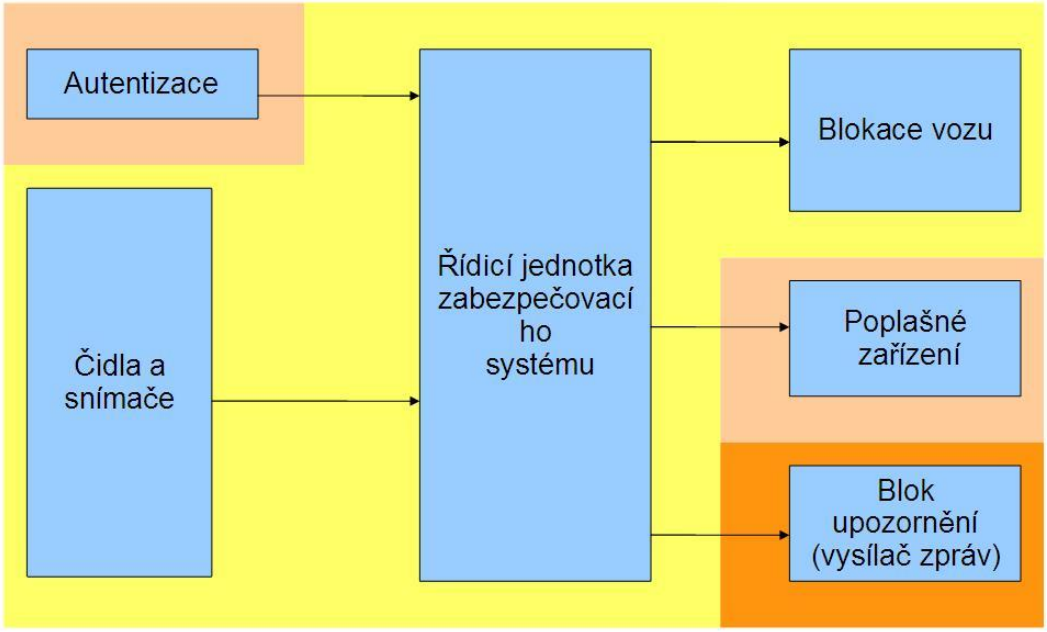
Kapitola 4

Návrh zabezpečení

Systém řeším mimo jiné s ohledem na rozšiřitelnost, modifikovatelnost a univerzálnost. Mou snahou je zajistit systém takový, aby bylo možné jej bez větších potíží užít v různých vozech. Dále chci, aby bylo možno vytvořit různé varianty - např. komplexní systém za vyšší cenu, nebo jednodušší levnější systém.

4.1 Blokové schéma systému

Rozdělil jsem tedy systém na několik dílčích bloků. Jejich uspořádání je patrné na obrázku 4.1, který ukazuje blokové schéma celého systému.



Obrázek 4.1: Blokové schéma

4.2 Čidla a snímače

Aby mohl systém vůbec nějakým způsobem pracovat, je nutné, aby dostával informace o tom, co se děje v hlídaném prostředí. Zde se jedná o automobil, a jak bylo uvedeno v části 3.2, základem je vniknutí do automobilu. Je nutné pohlídat si polohu zloděje. Venku vedle automobilu může být každý, kdo chce, tomu nemůžeme zabránit. Ovšem jakmile začne lupič na vozidlo sahat, začíná být nebezpečný a v okamžiku, kdy se dostane do vnitřního prostoru auta, vzniká závažný problém. Z toho plyne, že je třeba hlídat přístupové body. Těmi jsou:

- dveře
- kapota
- víko zavazadlového prostoru

4.2.1 Otvory

Je zřejmé, že se jedná o uzavíratelné otvory. Z toho plyne, že lze detekovat dva stavy : otevřeno a zavřeno. Je-li otvor zavřený, je to jedna z podmínek pro to, abychom mohli auto začít hlídat. Otevře-li se otvor v průběhu hlídání auta, je pravděpodobné, že došlo k nelegálnímu vniknutí a musíme vyhlásit poplach.

Proto je třeba vybavit dveře, kapotu i kufr vhodnými spínači. Nejjednodušší je mechanický rozpínací kontakt, kdy zavřené dveře tlačí na tlačítko, po otevření tlak zmizí a dojde ke spojení kontaktů - např. k ukostření (takto pracují např. kontakty ve Škodě Favorit, které slouží k ovládání osvětlení vnitřního prostoru). Tuto možnost považuji za nejlepší pro můj návrh.

Další možností jsou dveřní zámky složité elektromechanické konstrukce. Setkal jsem se s nimi v novějších automobilech pocházejících z koncernu Volkswagen Group, např. Škoda Superb nebo Volkswagen Passat. Jejich systém integruje do zámku centrální zamykání a snímače polohy zámku.

To jsou moje poznatky z občasných setkání s těmito automobily. Bohužel dokumentace k těmto vozidlům je dostupná jen v servisních sítích technikům a tak jsem nemohl více nastudovat.

4.2.2 Hýbání s vozem

Máme vyřešené dveře, co když do auta ale pachatel do auta vůbec nehodlá vstoupit? Jen si auto třeba vyheveruje a ukradne kolo. V takovém případě je třeba systému poskytnout informaci o tom, že k něčemu takovému dochází. K tomu je třeba použít náklonové a otřesové čidlo. Je zde třeba dbát na vhodné nastavení, automobil se může naklánět i vlivem silného větru, což není důvodem k poplachu.

4.2.3 Vnitřní prostor auta

Pokud by lupič vnikl do auta, a překonal při tom všechny předchozí nástrahy, měli bychom o jeho vniknutí mít informaci. Pohyb uvnitř vozidla bychom mohli detekovat světelnou závorou. To by ovšem nebylo vhodné vzhledem k příkonu i ke značně omezeným možnostem prostorového snímání. GES Electronics [5] má v katalogu několik závor, ale všechny vykazují značný příkon, zpravidla několik desítek mA. V praxi se užívá častěji ultrazvukového

snímače, který detekuje pohyb v prostoru, zatímco světelná závora zjistí pohyb jen při protnutí paprsku - přímky. V praxi jsem navíc zažil zajímavou situaci, kdy po uzamčení auta v garáži ultrazvukové čidlo rozšířilo svoje pole působnosti i na prostor garáže, zatímco při stání venku si hlídalo jen prostor v autě. Toto chování výrobce v dokumentaci sice neuváděl, ale bylo jednoznačně přínosem.

4.2.4 Navržené řešení

Blokové schéma 4.1 ukazuje Blok čidel. Jeho funkci jsem popsal v předchozím textu, teď ještě naznačím vnitřní strukturu. Elektrotechnické schéma konkrétní realizace je uvedeno v kapitole 5.1. Obrázek 4.2 ukazuje nutný základ, zatímco schéma 4.3 ukazuje spíše rozšíření. Zde je třeba uvést, co je blok logika, který se vyskytuje jak v 4.2 tak v 4.3. Na první pohled by se mohlo zdát, že se jedná o prostý logický součet - jakmile alespoň jeden ze snímačů zaznamená změnu, z logiky přejde signál do řídicí jednotky. Logický součet by byl použitelný jen v případě, že by všechna čidla hlásila v klidu logickou nulu a v případě narušení logickou jedničku. Pro účely této kapitoly postačí, že se jedná o hardware, který připraví informace tak, aby je mohl mikrokontrolér co nejlépe zpracovat.

4.3 Blokace vozu

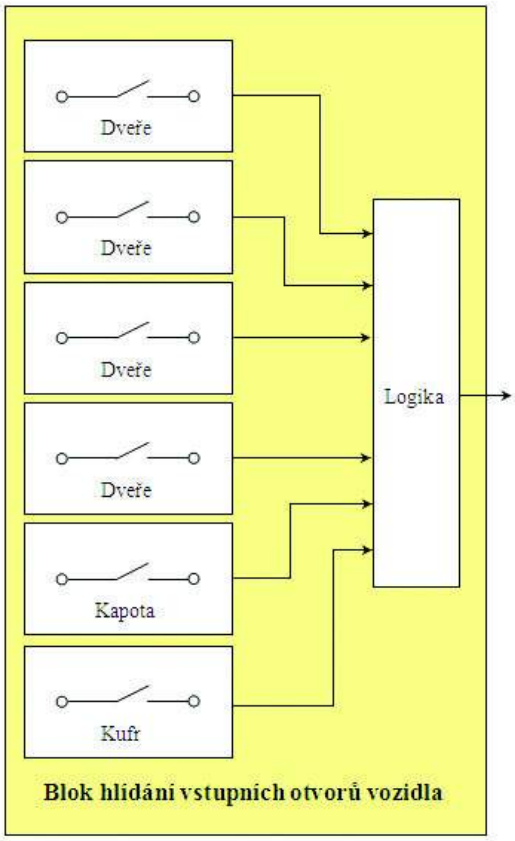
Blokace vozu je asi nejzásadnější věcí, je hlavním technickým cílem systému. Tak jako se snaží konstruktér znesnadnit vniknutí do vozidla, systém pro zabezpečení vozu se snaží vozidlo uvést do nepojízdného stavu. V kapitole 3.2.2 uvádím, příklady komponent automobilu, které je možné zablokovat. Princip je poměrně snadný. Prostě přerušíme napájení blokovaného prvku nebo dáme pokyn nějakému akčnímu členu. Velmi snadný a přitom bezvadně funkční způsob odpojování napájení ukazuje obrázek 4.4. Signál z řídicí jednotky ovládá rozpínací relé. To rozpojí silové kontakty, přes které je přivedeno napájení k blokovaným dílům, např. k palivovému čerpadlu apod. Akční člen může mechanicky zablokovat např. převodovku.

4.4 Autentizace

Pod pojmem autentizace rozumíme obvykle ověření pravosti uživatele. Já jsem ho v tomto případě převzal pro proces ověření, zda autem chce odjet, nebo jej jen otevírá ten, kdo na to má právo, tedy ten, kdo auto vlastní, nebo ten, kdo si jej půjčil, či ten, kdo na takovou činnost právo nemá. Naprostým základem je klíč. Ten kdo má správný klíč do zámku vozu, je nejspíše tím, kdo má právo s vozidlem manipulovat. Snímače v zámku nám řeknou, jestli je zámek odemknut či nikoli a to nám vlastně poví, zda je klíč správný - špatným odemknout nelze.

Jak již bylo uvedeno, může dojít k neoprávněnému otevření i bez klíče, o tom nám samozřejmě zámek neřekne nic, prostě je otevřen. Pro tyto případy je vhodné systém doplnit dalším kontrolním mechanismem.

1. Nejjednodušším může být skryté tlačítko, o němž pachatel neví. Takovým tlačítkem pošleme řídicí jednotce signál - jsem to já, odblokuj auto. Takové tlačítko může také sloužit pro odblokování v případě nouze, například při potížích s klíčem, zapomenutí kódu apod.



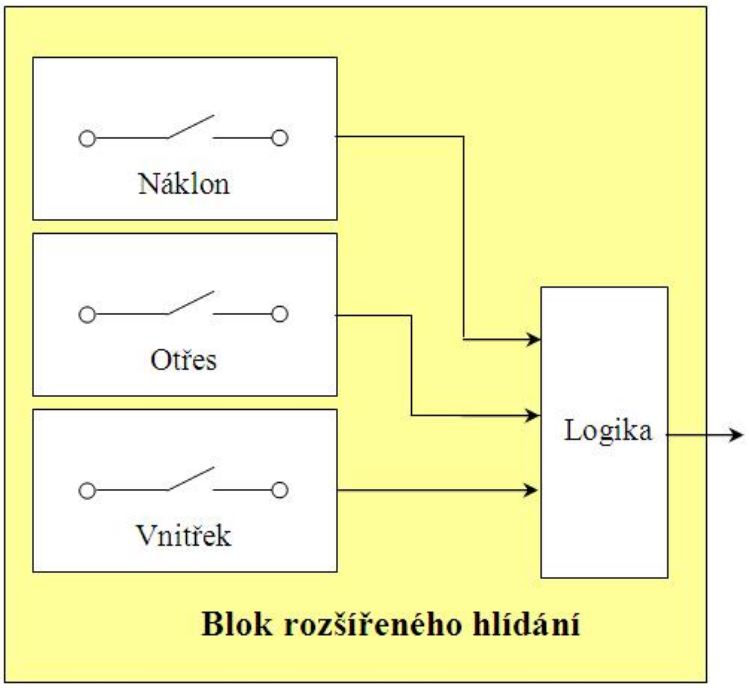
Obrázek 4.2: Blok hlídání vstupních otvorů vozidla

2. Pokročilejší systém bude obsahovat numerickou klávesnici s nutností vložení kódu PIN.

3. Další variantou je použití čipu v klíčku, a jeho čtení při otáčení klíčem ve spínací skříňce. Tento systém pracuje na magnetickém principu. Je téměř nemožné podvrhnout klíč, nastartovat jiným, například okopírovaným klíčem, protože čip v klíčku obsahuje kód, a tento kód je plovoucí.

4. Dálkové ovládání umožní vyslat autorizační údaje, samozřejmě v plovoucím kódu na dálku.

Možnosti 1, 2, 3 si bohužel příliš nerozumí s poplašným zařízením. Není dost dobře možné vyhlásit poplach po otevření dveří a obtěžovat s ním uživatele dokud nezadá správný kód, přinejmenším je vhodné ponechat časovou prodlevu, nicméně ani to nemusí být úplně vhodné. Dlouhá prodleva umožní nerušené vykradení auta, krátká bude otravovat tak dlouho, až uživatelům dojdou nervy a systém odpojí.



Obrázek 4.3: Blok rozšířeného hlídání

4.5 Poplašné zařízení a blok upozornění

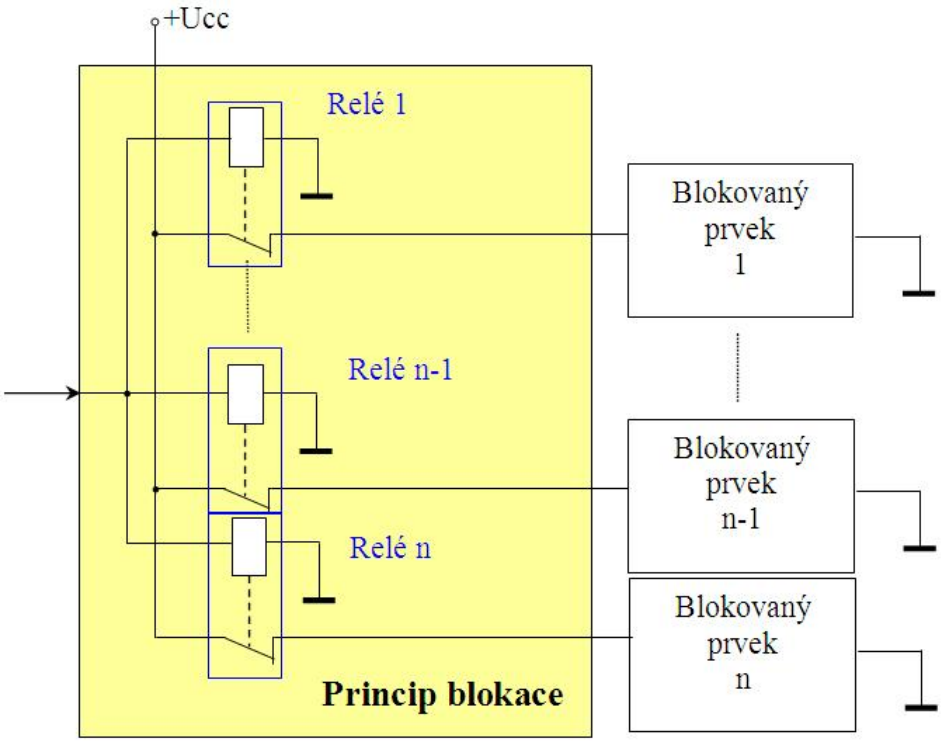
Poplašné zařízení bude pravděpodobně tím nejjednodušším blokem v celém systému. V případě vyhodnocení situace jako útok pošle řídicí jednotka signál a vypukne poplach. Začne houkat siréna, blikat výstražné světlomety, troubit klakson. Pokud by byl systém vybaven blokem upozornění, mohlo by dojít např. k odeslání sms majiteli.

Pokud jde o blok upozornění, dospěl jsem k závěru, že jde o problematiku jen těžko realizovatelnou v mých podmínkách. Proč? Vzhledem k dosahu dostupných technologií přichází v úvahu jedině GSM komunikace. Bezdrátové standardy WiFi, Bluetooth nebo ZigBEE komunikují jen na malou vzdálenost nevhodnou pro zasílání zpráv majiteli či bezpečnostní agentuře. Komunikace GSM má ovšem jeden zásadní problém - zasílání SMS či telefonování je zpoplatněno dle operátora a nemyslím si, že je důležité v tomto případě takovou komunikaci provádět a zbytečně platit, protože se nejedná o klíčový prvek systému.

4.6 Řídicí jednotka

Z blokového schématu 4.1 je zřejmé, že se jedná o centralizovaný systém. Informace o stavu auta zpracuje řídicí jednotka, která na jejich základě vydá pokyny ke správné akci.

Tato jednotka by mohla být realizována dvěmi způsoby - buď čistě hardwarově, pomocí kombinačních obvodů, nebo jako mikrokontrolér s vhodným softwarem. Systém sestávající ze sekvenčních obvodů by byl rychlý a jednoduše realizovatelný. Nicméně jeho značnou nevýhodou by byla nutnost napájet celou řadu integrovaných obvodů. Proto navrhuji naprogramování mikrokontroléru. Další výhodou tohoto řešení je schopnost použít mikrokontrolér i pro složitější aplikaci, mohl by zvládnout některé zajímavé věci popisované v závěru.



Obrázek 4.4: Princip blokace

4.6.1 Základní princip návrhu

Mikrokontrolér bude zpracovávat vstupní informace a odesílat výstupní. Na vstup bude dostávat informace podle blokového schématu 4.1 a informaci, zda někdo zamkl, nebo odemkl vozidlo. Ta poslouží vlastně jako vypínač bezpečnostního systému. Nepotřebujeme zabezpečovat auto, které je odemknuté (zde mám na mysli odemknutí normální, nikoli vloupání). V odemknutém stavu je třeba ještě zohlednit blok autorizace a standardně není důvod k blokování auta.

Mikrokontrolér by měl být v době hlídání auta v klidu, v režimu s nízkou spotřebou energie a probrat by se měl jen tehdy, dojde-li k útoku, nebo odemknutí auta.

Vývojový diagram 4.5 ukazuje činnost jednotky. Je zde důležité zejména rozhodování. Při realizaci se bude rozhodovat podle logických úrovní na příslušných vstupech mikrokontroléru.

4.6.2 Činnost

Jak je patrné z obrázku 4.5, systém musí neustále zjišťovat, co se s hlídaným automobilem děje, a na základě získaných dat provádět určité akce. Tedy:

- Je-li automobil uzamčený, musí systém detekovat poplachová hlášení a v případě detekce poplachu zablokovat vozidlo, spustit alarm a také vše zaznamenat do paměti systému.

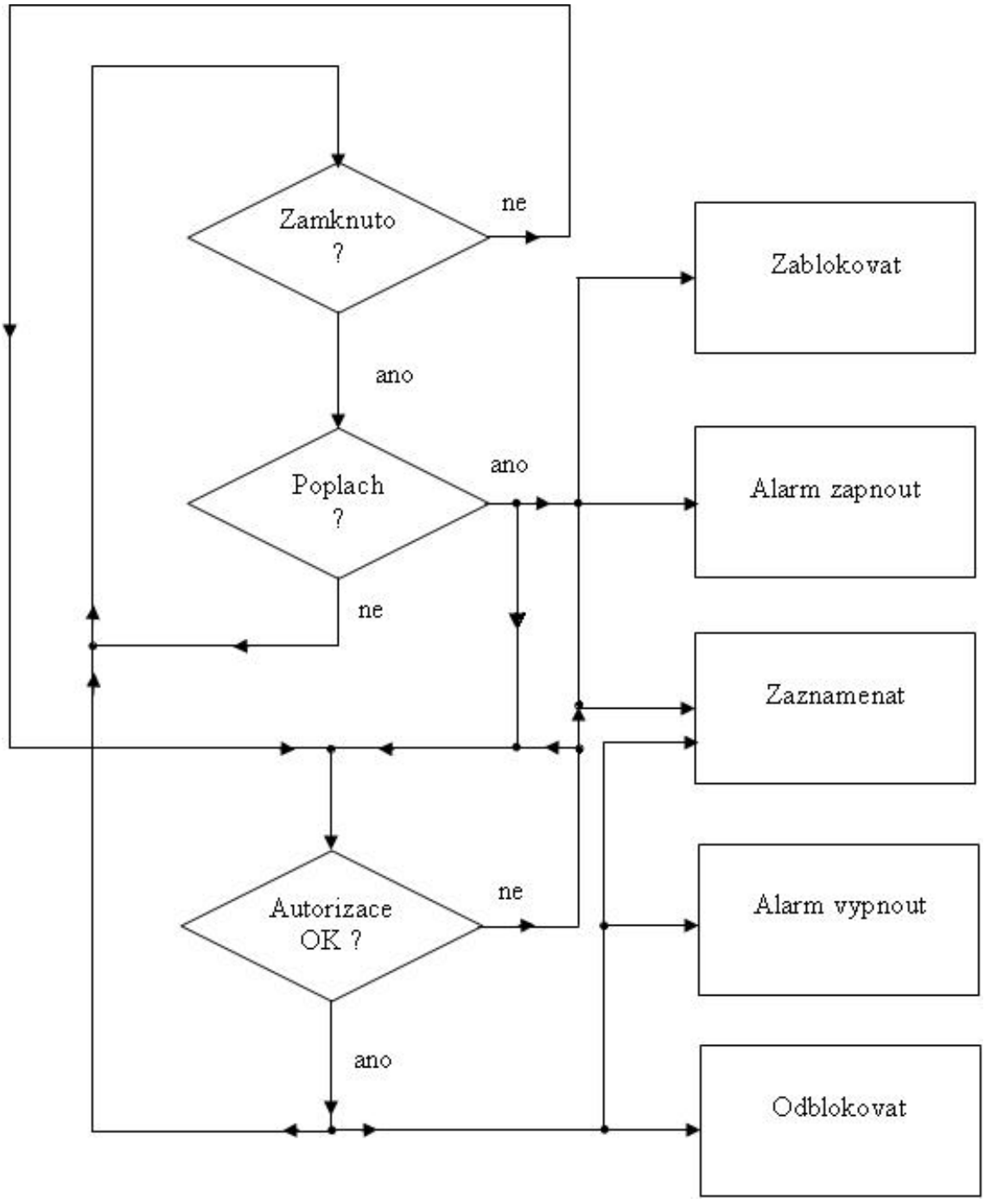
- Pokud se v případě poplachu uživatel autorizuje, poplach bude ukončen, vozidlo odblokováno a autorizace bude zaznamenána do paměti.
- Neplatná autorizace vyvolá poplach , zablokování vozidla a záznam do paměti.
- Po odemčení je testována autorizace
- Není-li vyhlášen poplach testuje se ve smyčce zda je vozidlo odemknuto.

4.7 Blokace vozu 2

Z grafu 4.5 je zřejmé, že blokace nastává až v okamžiku vyhlášení poplachu. Na první pohled by mohlo být lepší blokovat automobil v okamžiku uzamčení.

Znamenalo by to, že by se s každým uzamčením a odemčením provádělo odpojování i připojování zařízení, případně by docházelo k aktivaci akčních členů. Vždy by se jednalo o nějakou činnost elektromechanických zařízení. A tady by mohl vzniknout problém se spolehlivostí vzhledem k opotřebení a mohlo by dojít zbytečně ke situaci, kdy prostě neodjedeme, protože se opotřebil některý z kontaktů.

Přitom blokací uzamčeného automobilu nic nezískáme. Zablokovat jej potřebujeme až když se něco děje. A protože se něco děje jen výjimečně, zvýší blokace až při poplachu spolehlivost systému.



Obrázek 4.5: Princip činnosti

Kapitola 5

Realizace

5.1 Hardware

Pro realizaci bylo třeba zvolit nějaký mikrokontrolér. Uvažoval jsem, že použiji mě známých prvků. To by znamenalo použití mikropočítače z řady 8051 nebo podobného, či Motoroly řady HC08. S oběma tolik odlišnými typy jsem se už prakticky setkal. Protože programování 8051 není vůbec náročné, chtěl jsem více použít tu. Avšak vedoucí práce pan Ing. Josef Strnadel, PhD., přišel s návrhem využití FITkitu [12]. Nemusel mě dlouho přemlouvat a souhlasil jsem.

5.1.1 Proč FITkit

FITkit je hardwarová platforma obsahující mikrokontrolér TI MSP430 a FPGA Xilinx Spartan 3 [12]. Jeho výhodou oproti dalším uvažovaným řešením je komunikace s PC a programování prostřednictvím USB [12]. Není tak potřeba používat speciální programovací přípravky - programátory. To velmi ovlivnilo moje rozhodování. Dalším významným důvodem, proč jsem zvolil FITkit je i to, že ač jsem o něm už mnoho slyšel, doposud jsem jej nepoužil. Myslím, že je vynikající naučit se něco nového.

5.1.2 Elektrická část

V okolí mikrokontroléru se musí nacházet elektronické obvody, které budou s mikrokontrolérem nějakým způsobem komunikovat a zasahovat do instalace vozidla.

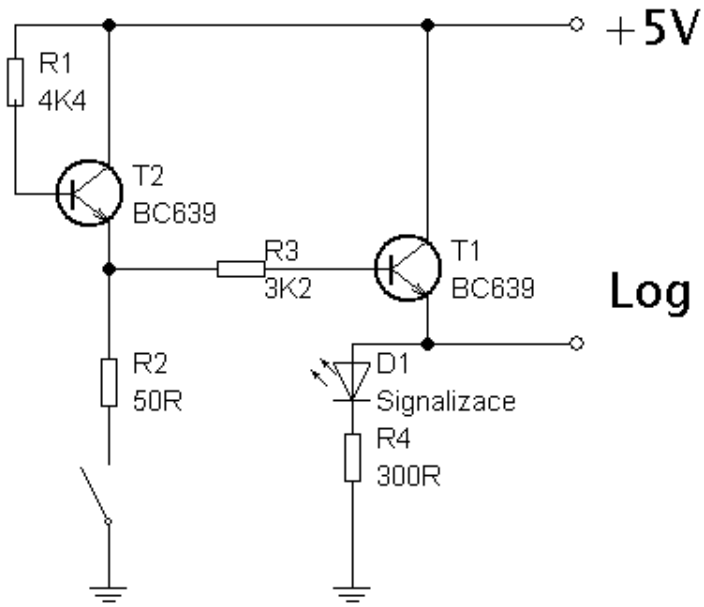
Integrované obvody

Určité funkce bude výhodné realizovat mimo mikrokontrolér pomocí integrovaných obvodů. Vzhledem k potřebě nízkého příkonu jsem se rozhodl pro obvody využívající technologii CMOS. Technologii CMOS jsem studoval mimo jiné i v knize Polovodičové paměti a jejich použití [3] . Jednou z vlastností CMOS obvodů je jejich minimální až zanedbatelná spotřeba v klidovém stavu, daná pouze svodovým proudem zavřeného unipolárního tranzistoru a jsou tedy vhodné tam, kde je třeba dosáhnout nízké spotřeby.

Spínaná kostra

Spínače ve dveřích či na kapotě spínají kostru. Znamená to, že ke spínači vede vodič, který se při otevření dveří ukostří, jinak končí ve vzduchu. Tento stav je nevhodný pro

detekci pomocí logických obvodů, které vyžadují zřetelné napěťové úrovně. Proto jsem navrhl obvod, jehož funkcí je převod signálu uzemněn/neuzemněn na napěťové úrovně. Princip převodu ukazuje schéma na obrázku 5.1.



Obrázek 5.1: Princip převodu spínané kostry na napěťové úrovně

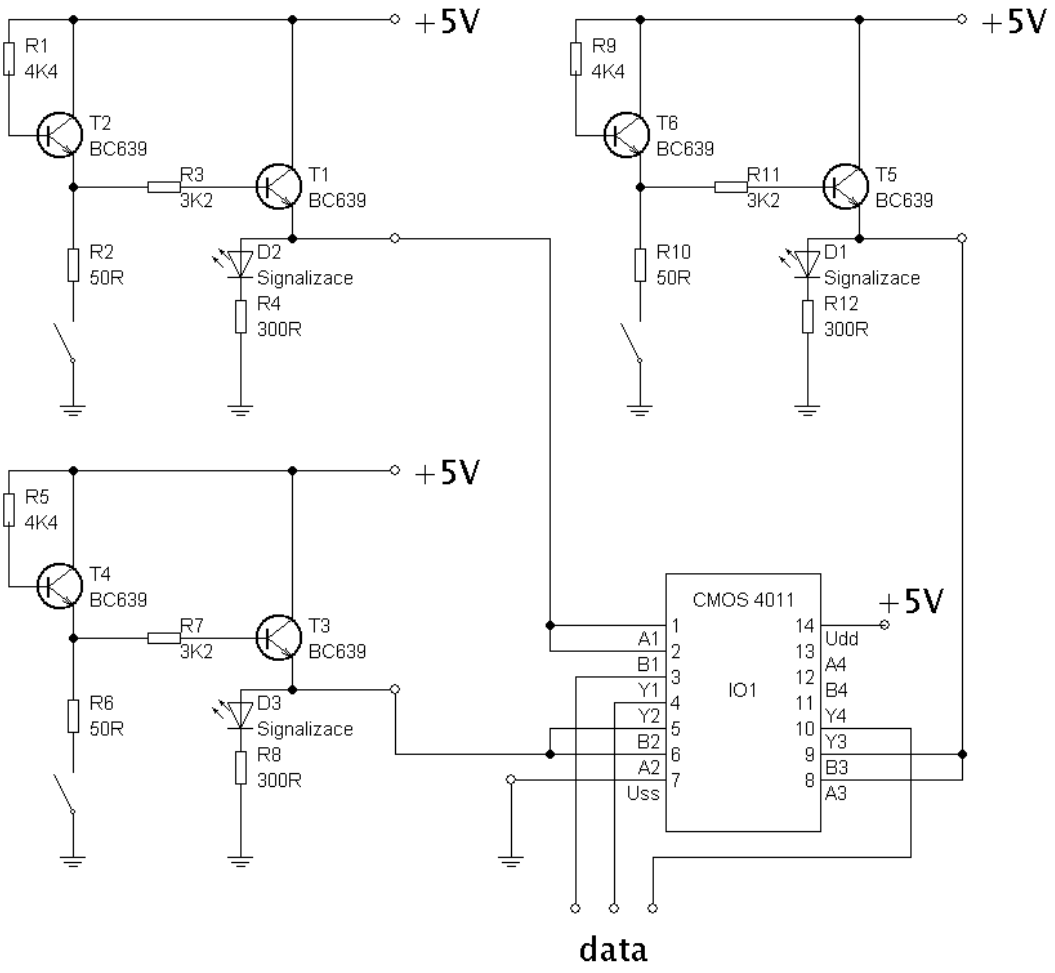
Oba dva tranzistory jsou technologie NPN, tedy pro jejich otevření je třeba připojit na bázi kladné napětí. Spínač připojený k emitoru tranzistoru T2 představuje diskutované spínání kostry. Tranzistor T2 je vždy otevřený. Je-li kostra odpojena, naměříme na emitoru proti zemi napětí dané vstupními hodnotami tranzistoru a jeho zesílením, tedy maximální, které tranzistor pustí. Naopak, je-li kostra připojena, napětí naměříme nízké, bude se jednat vlastně jen o úbytek na rezistoru R2. Tranzistor T1 toto napětí ještě zesílí a na jeho emitoru naměříme použitelné logické úrovně (ve schématu označeno Log) .

Toto napětí už bez potíží zpracují logické obvody CMOS s napájením 5V. Pro zkvalitnění signálu jej ještě připojím na vstupy obvodu CMOS 4011 (4×dvouvstupový NAND), kterým úrovně upravím na precizní logickou jedničku a nulu.

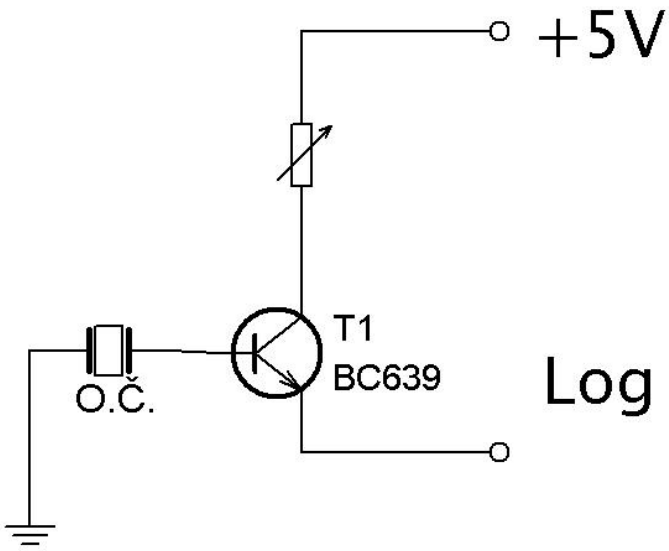
Realizované zapojení spínané kostry

Schéma na obrázku 5.2 ukazuje skutečně realizované zapojení spínačů v modelu systému. Je vidět realizace obvodu ze schématu 5.1. Z obvodu CMOS 4011 pak získávám informaci o stavu spínačů.

Realizoval jsem tři spínače. Tento počet jsem zvolil proto, že další spínače, kterých by teoreticky mohlo být mnohem více, by pouze kopírovaly zapojení prvního a nic nového by to pro model už nepřineslo. Méně spínačů by zase neukázalo možnosti jejich identifikace a zaznamenávání hlášení.



Obrázek 5.2: Realizované zapojení



Obrázek 5.3: Připojení otřesového snímače.

Čidlo nárazů a vibrací

Čidlo nárazů a vibrací funguje při detekování nárazu jako zdroj napětí. Na výstupu se objevuje v závislosti na síle nárazu a frekvenci vibrací napětí. Toto napětí jsem připojil do báze tranzistoru. Na jeho kolektoru je napájecí napětí připojené přes sadu rezistorů, emitor je pak připojen k logice CMOS podobně jako spínače na kostru.

Sada rezistorů v kolektoru umožňuje v modelu ladění obvodu, použitelných výsledků jsem dosáhl s hodnotou $12k\Omega$ výhodnější by bylo použití potenciometru, tak jak je to nakresleno ve shématu 5.3. Uvedené zapojení je experimentální a pro praktické použití by bylo třeba najít přesnější nastavení a možná i úpravu množství a zapojení okolních součástek. Výstup Log pak projde dvojicí obvodů CMOS NAND a je použitelný v mikrokontroléru. Čidlo tedy používám digitálně, otřesy jsou, nebo nejsou. Nejsou znamené v tomto případě otřesy, které otevřou tranzistor tak, že na jeho emitoru je napětí nižší, než je rozhodovací úroveň pro log. 1. To je v tomto případě cca 2,5V. Pravděpodobně by šlo využít i AD převodník.

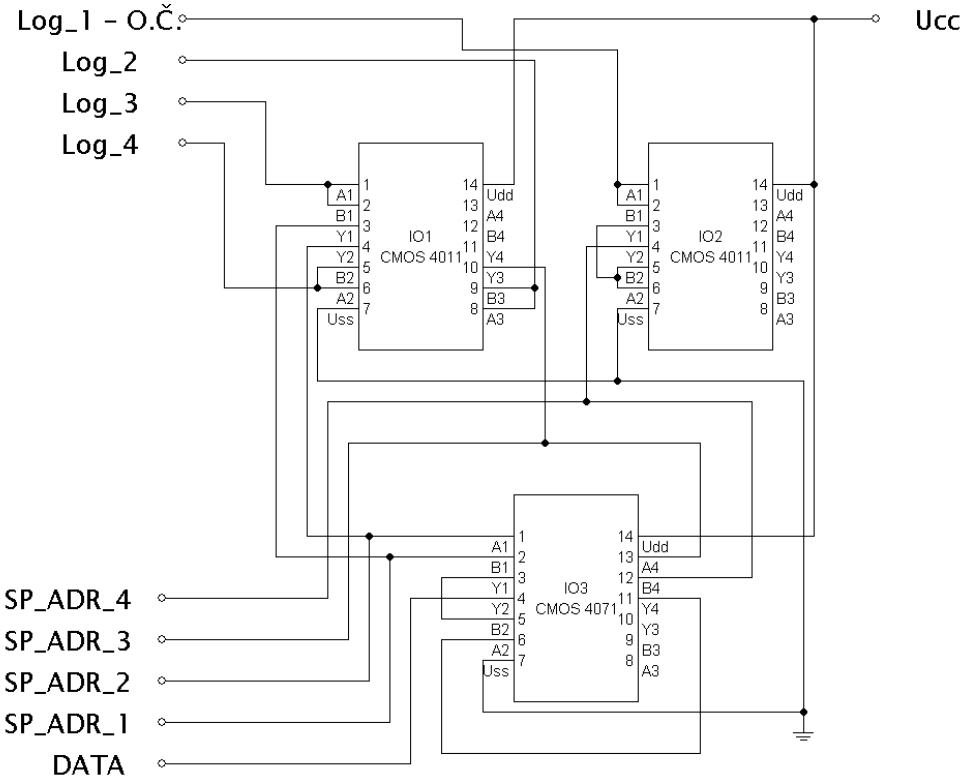
Obrázek 5.4 představuje realizaci logické části popsané v části 5.2.2. Vstupy v horní části představují spojení pro čidla a spínače zapojené podle obrázků 5.3 a 5.1.

Piezoelektrický bzučák

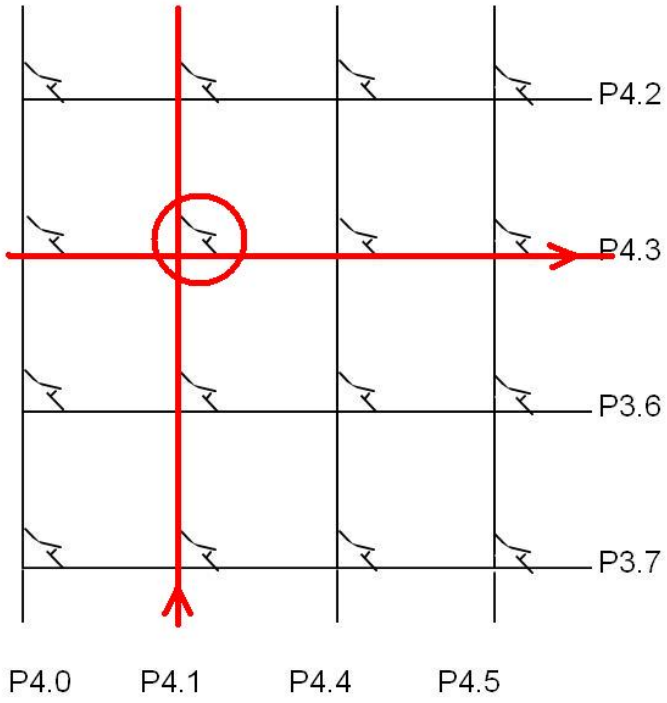
Bzučák 5.3.1 spínám přímo logickou jedničkou na portu mikrokontroléru.

5.1.3 Reálný čas

Pro záznam dat do paměti systému je třeba vybavit systém hodinami tak, aby u každého záznamu bylo uvedeno kdy k akci došlo. Hardwarový obvod reálného času není v MSP430F168 k dispozici, proto je třeba vytvořit softwarový.



Obrázek 5.4: Schéma logiky systému.



Obrázek 5.5: Připojení klávesnice k mikrokontroléru.

Principem je vytvoření proměnné, jejíž hodnota bude pravidelně inkremetována tak, abychom mohli jednoznačně určit, jaký počet inkrementací je potřebný pro změnu reálného času o nejmenší rozlišovanou jednotku. V tomto případě postačí rozlišení jednotek sekund.

Periodické činnosti lze dosáhnout s využitím časovače v režimu výstupní komparace. Přetečení časovače vyvolá přerušení a obslužná rutina upraví hodiny.

5.1.4 Klávesnice

Klávesnice je standardně součástí fitkitu a k mikrokontroléru je připojena pomocí FPGA, nicméně po dohodě s vedoucím práce jsem ji demontoval a připojil k mikrokontroléru pomocí konektoru JP9 [1], 5.3.3, pincip připojení ukazuje obrazek 5.5 a diskutuje v kapitole Software část 5.2.8.

5.2 Software

Tato kapitola popisuje konkrétní postupy realizace včetně ukázek částí zdrojových kódů tak, aby vystihly podstatu funkcí. Nejedná se o kompletní popis zdrojového kódu.

5.2.1 Hlavní smyčka

Mikrokontrolér v klidu, kdy je auto uzamčeno a není hlášen žádný poplach nepotřebuje, vykonávat žádnou zvláštní činnost. Jediné, co musí pracovat neustále, jsou hodiny.

Aplikace je z větší částí závislá na práci s příznaky. Pojmem příznak označuji v této aplikaci proměnné používané v softwaru, které některá z funkcí nastaví do potřebné hodnoty a jiná funkce tyto proměnné použije, případně změní. Tento přístup mi umožnil pohodlně přidávat postupně novou a novou funkčnost. Z toho důvodu považuji aplikaci za rozšiřitelnou.

5.2.2 Sledování

Jak bylo uvedeno dříve, sledování stavu střeženého automobilu probíhá pomocí soustavy snímačů a čidel. Protože chceme uchovávat informace o tom, který prvek způsobil vyhlášení poplachu, musíme vhodným způsobem zajistit detekci jeho adresy. Zvažoval jsem následující způsoby:

- Přímé připojení všech čidel/spínačů k mikrokontroléru
- Připojení čidel/spínačů k mikrokontroléru prostřednictvím dekodéru.

Použití dekodéru jsem nakonec zavrhl, protože by znamenalo navíc:

- Zvýšení ceny systému vzhledem k použití dalšího hardware
- Zvýšení spotřeby elektrické energie - další obvod s nutností napájení
- Zvýšené požadavky na obslužný software

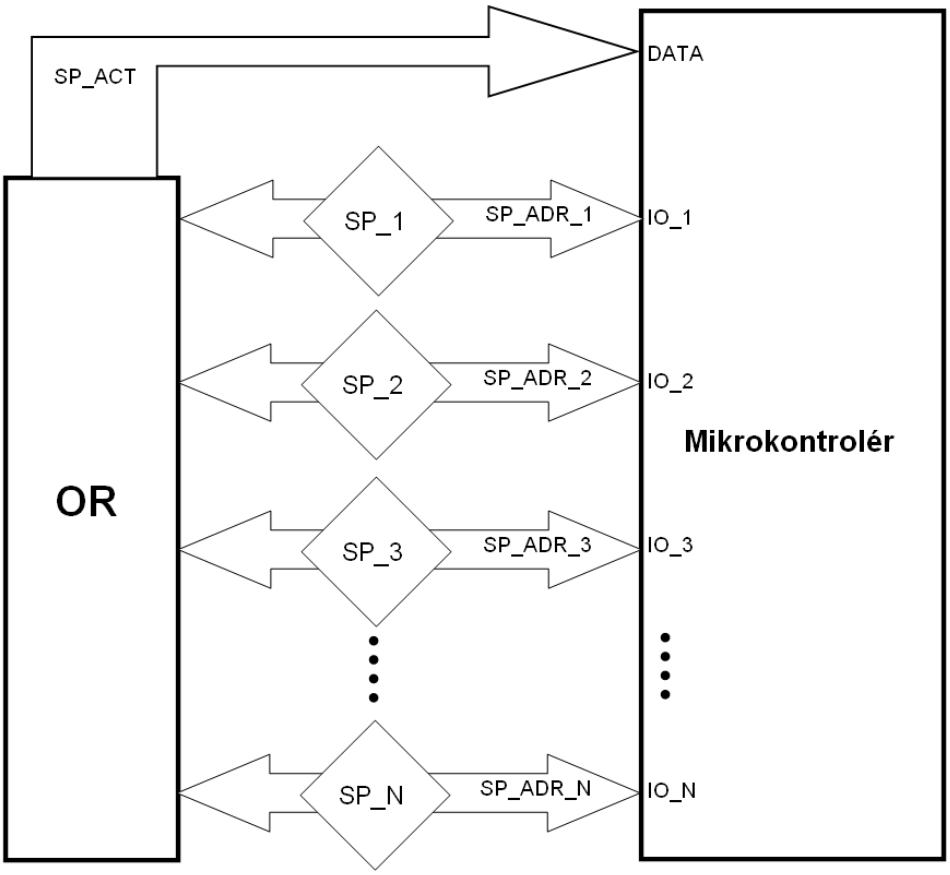
Přímé připojení hlídacích prvků ovšem také není ideální, protože představuje nutnost sledovat stav hned několika vstupů mikrokontroléru zároveň. V případě, že by bylo např. běžné pětidvéřové vozidlo typu hatchback plně vybaveno, znamenalo by to 5×dveřní spínač, 1×kapotový spínač, 1× ultrazvukový snímač a 1×otřesový snímač. To je celkem 8 vstupů, jejichž stav by byl třeba neustále detekovat, případně 8 vnějších přerušení. To je příliš, postupná detekce by neustále zaměstnával mikrokontrolér a tolik vnějších přerušení napatří mezi běžnou výbavu mikrokontrolérů. Navíc s případným rozšiřováním by velmi vzrůstala složitost systému.

Připojení jsem proto realizoval podle obrázku 5.6. Všechny spínače jsou spojeny pomocí logického součtu a přivedeny na jeden ze vstupů mikrokontroléru. Zároveň jsou tyto snímače připojeny samostatně, každé na jeden vstup mikrokontroléru.

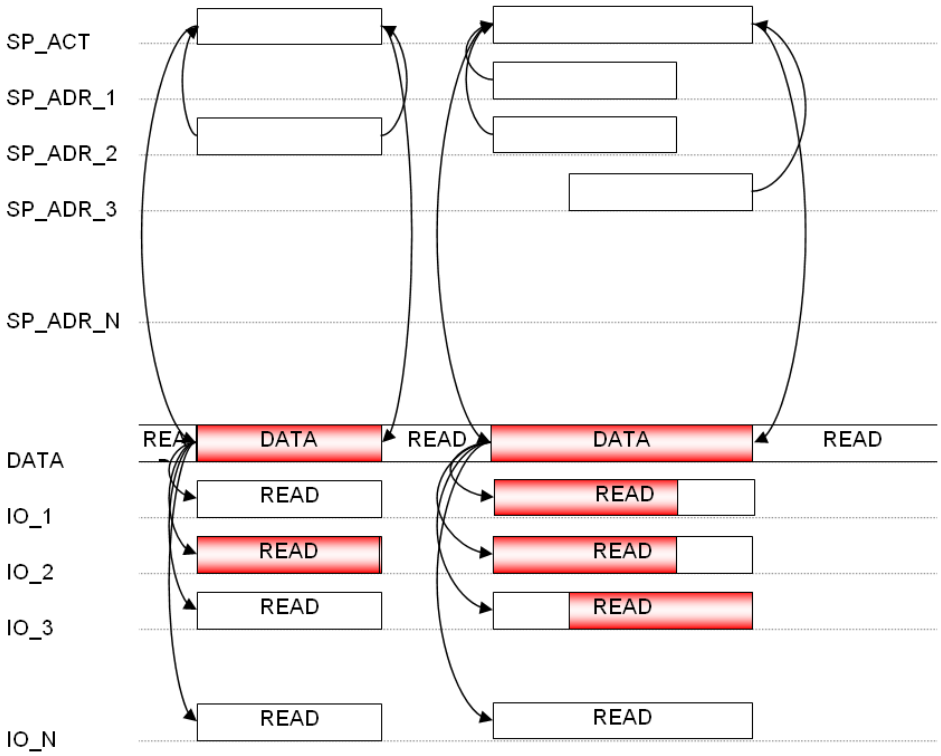
Činnost zapojení na obrázku 5.6 je zřejmá z diagramu 5.7. Mikrokontrolér se zajímá pouze o signál pojmenovaný DATA. Ten je vyveden z logického součtu. Jakmile se některý ze spínačů začne projevovat jako aktivní, logický součet nastaví svůj výstup také jako aktivní a na signálu DATA se objeví logická úroveň znamenající pro mikrokontrolér informaci, kterou mohu popsat slovy jako "Máme poplach, podívej se kde !" . Na základě toho mikrokontrolér začne číst všechny svoje vstupy, ke kterým jsou připojeny spínače, a získá tak jednu, případně i více adres, kde nastal poplach. Pokud přestanou spínače problém hlásit, logický součet (DATA) přestane být aktivní, a mikrokontrolér přestane číst adresy.

5.2.3 Archivace a zasílání dat

Systém nepoužívá velké množství dat, přesto mohou být tato data zajímavá. Pracuje se s hodinami a datem, pracuje se s adresou poplachu a pracuje se s identifikací uživatele. Všechna tato data mohou posloužit např. při diagnostice systému - např. v situaci, kdy bude systém zbytečně hlásit poplach můžeme v datech najít, který prvek poplach vyvolává a způsobit nápravu (např. výměnu vadného spínače).



Obrázek 5.6: Systém propojení čidel/spínačů s mikrokontrolérem



Obrázek 5.7: Grafické znázornění sledu událostí při aktivitě čidel/spínačů

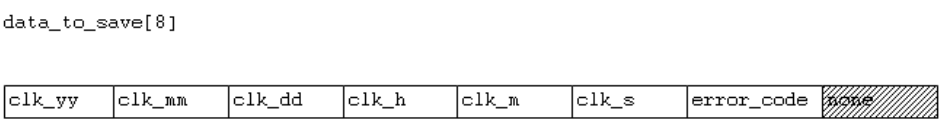
Pro ukládání dat jsem využil na FITKITu umístěnou SDRAM. Na první pohled by mohla vadit závislost sdram na napětí, nicméně zabezpečovací systém je a musí být pod napětím neustále, jinak nemůže fungovat.

Pro práci s pamětí jsem vyšel z příkladu na webu FITKITu [13]. Přepracoval jsem software pro MCU, ale vyžil jsem konfiguraci pro FPGA.

Základem jsou dva bloky, dvě funkce - jedna data ukládá do paměti, druhá je čte, pokud si to uživatel přeje - zašle data přes komunikační rozhraní do PC. S tím souvisí podpůrná funkce, která připraví data pro uložení do paměti.

Příprava dat

Definoval jsem pole, které slouží jako přípravné a v případě potřeby se jeho obsah uloží do paměti SDRAM. Způsob uložení dat je znázorněn na obrázku 5.8. Poslední prvek pole jsem nechal nevyužitý, jako rezervu pro budoucí rozšíření. Položky v poli jsem nechal pojmenované tak, jak jsou pojmenované proměnné v aplikaci. Jejich popis uvádí tabulka 5.2.3.



Obrázek 5.8: Způsob uložení dat v poli

označení	význam	poznámka
`clk_yy`	rok	pouze poslední dvojčíslí letopočtu
`clk_mm`	měsíc	
`clk_dd`	den	
`clk_h`	hodina	
`clk_m`	minuta	
`clk_s`	sekunda	
`error_code`	chybový kód	kód jedinečný pro každou událost
`|none|`	nula	místo ponechané pro případné rozšíření

Tabulka 5.1: Popis pole s daty připravenými pro uložení

Čas a datum v poli v poli aktualizuje přímo blok reálného času průběžně, chybový kód nastaví blok detekce poplachu 5.2.5, nebo blok autorizace.

Ukládání dat

Ukládání dat do paměti řeší funkce `save_to_ram` . Princip je ten, že funkce projde připravené pole 5.2.3 a nahraje v něm uložená data do paměti SDRAM. S tím souvisí adresy v paměti, na kterou se data budou ukládat a počet záznamů, které budeme uchovávat. Paměť je velká 8MB, jeden záznam představuje 8 položek, každou o velikosti 1B. Kapacita paměti je tedy více než dostatečná.

Nabízí se otázka, kolik dat má smysl uchovávat. Jeden záznam nám moc neřekne, milion záznamů bude znamenat nepřeberné množství dat, o jejichž smyslu nejsem přesvědčen. K

čemu by byl dobrý záznam, že někdy v minulosti vyhlásilo poplach vnitřní čidlo? Myslím si, že k ničemu, proto jsem se rozhodl množství záznamů omezit na 1000 položek. Až se 1000 položek zaznamená, začnou se od začátku přepisovat v kruhu, zůstane tak k dispozici stále 1000 použitelných posledních záznamů.

Adresování v paměti je řešeno složením počáteční adresy plus po každém záznamu přepočítávaný index, využívám je i ve funkci čtení dat 5.2.3

Tabulka chybových kódů

Význam	Označení v aplikaci	Kód
levé přední dveře	`LEFT_FRONT_DOORS`	1
pravé přední dveře	`RIGHT_FRONT_DOORS`	2
levé zadní dveře	`LEFT_REAR_DOORS`	3
pravé zadní dveře	`RIGHT_REAR_DOORS`	4
kapota	`HOOD`	5
páté dveře	`FIFTH_DOORS`	6
vnitřní prostor	`INTERIOR`	7
otřesový snímač	`VIBRATION_SENSOR`	8
neoprávněná autorizace	`INVALID_AUTHORIZATION`	50
uživatel 1	`USER1`	101
uživatel 2	`USER2`	102
uživatel 3	`USER3`	103

Tabulka 5.2: Tabulka chybových kódů

Čtení dat

Čtení a zasílání dat realizuje funkce `read_ram`. Načte všechny záznamy v SDRAM a pošle je na terminál. Zde je prostor pro případné rozšíření, místo terminálu by mohlo data požadovat další zařízení připojené přeš RS232 a zpracovat je dle potřeby.

Funkce `read_ram` projde paměť od začátku až do posledního záznamu a 4ten8 data posílá na terminál. Informace o posledním záznamu nastavuje funkce pro zápis 5.2.3 . Funkce se volá uživatelsky.

5.2.4 Hodiny

5.1.3

Při vytváření hodiny jsem vyšel ze základu na webu [10]. Oproti původní aplikaci jsem zejména odstranil zobrazování dat na displeji, doplnil jsem práci s datem. S tím souvisí i problém přestupných roků a příprava dat pro ukládání do paměti 5.2.3. Zároveň nechávám pravidelně při každém přerušení testovat signál `DATA`, viz. 5.2.5.

Hodiny tedy řeší přerušení od časovače. Konfigurace časovače a přerušení proběhne v hlavní smyčce programu, zbytek řeší obslužná rutina přerušení.

Pracuje se s proměnnými podle tabulky 5.2.4. Proměnné jsou globální, přistupují k nim i funkce pro nastavování času a data.

označení	význam
clk_yy	rok
clk_mm	měsíc
clk_dd	den
clk_h	hodina
clk_m	minuta
clk_s	sekunda

Tabulka 5.3: Popis pole s daty připravenými pro uložení

Konfigurace

Časovač běží v režimu výstupní komparace. Konfigurace je převzata z příkladu [10] Povolení přerušení časovače způsobí kód `CCTL0 = CCIE;` . Nastavení hodnoty komparace provede `CCR0 = 0x8000;` .

Obslužná rutina

Obslužná rutina invertuje každou jednu sekundu zelenou LED v dolním rohu FITkitu, tuto vlastnost jsem ponechal aktivní jako signalizaci chodu. V praxi většinou v autě také nějaká LED bliká, zpravidla červená. Další funkcí je test signálu

Dále proběhne navýšení počtu sekund, pokud přetečou stanovený limit, tj. 60, zvýší se počet minut a sekundy se nulují atd. Stejným způsobem se upravuje i datum. Zde ale vzniká problém s přestupným rokem a s nestejným počtem dní v měsících.

Další významnou činností obslužné rutiny je volání funkce `void test_DATA (void)` a příprava dat pro uložení do pole.

```
interrupt (TIMERA0_VECTOR) Timer_A (void)
{

 ...

  clk_s++;

  if (clk_s == 60)
  {
    clk_m++;
    clk_s = 0;
  } // kazdou 60. sekundu inkrementuj minuty


  .....


  //priprava dat k ulozeni
  data_to_save[0] = clk_yy;
  data_to_save[1] = clk_mm;
  data_to_save[2] = clk_dd;
```

```
  data_to_save[3] = clk_h;
  data_to_save[4] = clk_m;
  data_to_save[5] = clk_s;


  test_DATA ();


}
```

Činnost funkce `void test_DATA (void)` vysvětluje část 5.2.5, ukládání dat projednává část 5.2.3.

Problém přestupného roku

Problém přestupného roku se řídí podle pravidel [15]

1. Rok je přestupný, pokud je dělitelný číslem 4 (1996, 2004)
2. Výjimka č. 1 : Rok není přestupný, pokud je dělitelný číslem 100 (1900, 2100)
3. Výjimka č. 2 z výjimky č. 1: Rok, na který se vztahuje výjimka č. 1, je přestupný, pokud je dělitelný číslem 400 (2000, 2400)
4. O dalších pravidlech či výjimkách se rozhodne podle potřeby

Implementoval jsem tedy tento algoritmus, až na bod číslo 4, protože mi žádná další výjimka není v současnosti známa.

Letní čas

Letní čas je označení systémové úpravy měření času, při které se v letních měsících roku nepoužívá čas daný příslušným časovým pásmem, ale používá se čas, který je o určitou hodnotu (obvykle o 1 hodinu) posunut dopředu.

Cílem letního času je úspora elektrické energie, která by byla jinak potřeba pro večerní osvětlení. Důvodem je, že většina lidí je aktivnější večer (po západu slunce) než ráno (před východem slunce). [18] .

Bylo by nepohodlné až možná hloupé, kdyby musel uživatel ručně přepínat čas při změně hodin, tedy dvakrát do roka. Systém musí pracovat tak, aby neobtěžoval, ale pomáhal. Proto je třeba, aby byl systém tuto změnu provést automaticky.

Pro účely této práce je důležité datum, kdy se čas mění. To není fixní, např. 21.září, ale plovoucí tak, aby ke změně došlo vždy o víkendu, tedy v době, kdy není obvykle pracovní den. Wikipedie [18] o tom říká, že na letní čas se v ČR každý rok přechází poslední neděli v březnu, kdy se z 01:59:59 SEČ (středoevropského času) posunou hodiny na 03:00:00 SELČ (středoevropského letního času). Letní čas končí poslední neděli v říjnu, kdy se z 02:59:59 SELČ hodiny posunou na 02:00:00 SEČ. Toto uspořádání platí od roku 1996, do té doby byl konec letního času obvykle poslední neděli v září.

Z hlediska softwaru je tedy potřeba najít poslední neděli v březnu a poslední neděli v říjnu. Bude tedy nutné, aby nastavení systému obsahovalo zároveň i zadání, jaký je aktuální

PO	ÚT	ST	ČT	PÁ	SO	NE
			01	02	03	04
05	06	07	08	09	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	31	

Tabulka 5.4: Kalendář 1

PO	ÚT	ST	ČT	PÁ	SO	NE
				01	02	03
04	05	06	07	08	09	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	31

Tabulka 5.5: Kalendář 2

den. Potom můžu každý další den upravit den na aktuálním tedy po sobotě neděle, po neděli pondělí, atd.

Když znám aktuální den, potřebuji zohlednit datum. Kalendáře 1 a 2 ukazují dva možné extrémy - aktuální měsíc nedělí končí, nebo měsíc následující nedělí začíná a měsíc aktuální tedy končí sobotou. Je zřejmé, že poslední neděle v měsíci může nastat mezi 25. a 31. dnem včetně. Březen i říjen jsou měsíce o shodné délce 31 dní.

Tedy v případě, že aktuální datum je někde mezi 25.a 31. říjnem nebo březnem, software detekuje , který je den a v případě neděle posune hodiny odpovídajícím způsobem.

Funkce automatické úpravy času na letní však nemusí být vždy žádoucí, protože v některých zemích se čas nemění. Proto jsem umožnil její aktivaci či deaktivaci uživatelem.

5.2.5 Detekce poplachu

Detekce poplachu je řešena funkcí `void test_DATA (void);`, která přečte hodnotu signálu a zpracuje dle 5.2.2 . Každou sekundu tak dojde při přerušení k testu. Během jedné sekundy než dojde k dalšímu přerušení tak má potenciální pachatel tak teoretickou šanci během automobil otevřít a vyřadit systém z provozu. Tato šance je ale naprosto nereálná.

unkce `void test_DATA (void)` představue vlastní testování, vstupů. Nejprve se testuje, je-li auto zamknuté, odemknuté nehlídáme. Pokud je zamknuto, ulozi se hodiny pro `void poplach(void)` a probiha testovani, zda neni někde hlaseni poplachu v souladu s grafem 5.7.

```
void test_DATA (void)
{

  zamknuto_test (); //je zamknuto ?
  if (zamknuto && P6IN & 0x08)
  { //tady je signal data P6.3
```

```
    if (!general_action)
    { //ulozime hodiny, ale pouze pro prvnim vyhlaseni poplachu

      if (clk_s == 0)
      { //pokud bychom ukladali pri nula sekundach, doslo by k chybe
        time_alarm = 59;
      }
      else
      {
        time_alarm = (clk_s - 1);
      }

    }

    if (!autorizace_OK)
    {
      general_action = 1;
    }

    if (P6IN & 0x80)
    { //tady je signal ADRESY  NA P6.7

      error_code = HOOD; //ulozeni chuboveho kodu
      save_to_ram (); //ulozeni dat do pameti
  ...


}
```

5.2.6 Reakce na poplach

Nastane-li poplach, je třeba, aby systém vykonal nějakou činnost. Realizoval jsem:

- Rozblikání výstražného světla
- Aktivaci sirény
- Obecnou akci

Zatímco první dvě akce jsou zřejmé, pod pojmem obecná akce jsem skryl přípravu pro obecně libovolné rozšíření systému. Jedná se o nastavení globální proměnné `general_action` do hodnoty jedna v případě poplachu, jinak je tato proměnná nastavena na nulu. Jedná se vlastně o příznak, který je čitelný ostatním funkcím a lze tak libovolnou funkci naprogramovat jako reakci na nastavení této proměnné.

Rozblikání výstražného světla předpokládá hardwarovou podpdporu vozidla, tzv. přerušovač, aktivace sirény je díky použitému hardwaru 5.3.1 možná jen nastavením logické úrovně na výstupu mikrokontroléru.

S problematikou rozblikání výstražného světla a aktivací sirény souvisí ještě jeden problém. Výstražnými světly budou v praxi směrová světla automobilu a sirénou nebude modelový piezo bzučák, ale větší zařízení - např. Piezoelektrická siréna SA-105 z nabídky

Jablotronu [8]. Zmiňovaná siréna má odběr 250mA. Směrová světla, nejsou-li vytvořena moderní LED technologií, bývají tvořena 4 žárovkami s příkonem 21W a dvěma 5W kusy. To je v součtu téměř 100W, což odpovídá proudu o přibližné velikosti 8A. Budu-li uvažovat průměrný akumulátor s kapacitou 55Ah, byl by po sedmi hodinách akumulátor zcela vybitý. Je ale nutné počítat s tím, že světla nesvítí, nýbrž blikají. Mohu tak odhadnout poloviční spotřebu, i tak je ale velmi vysoká. Proto jsem blok poplachu realizoval tak, že výstražné signály budou po vyhlášení poplachu aktivní po dobu 59s, potom dojde na 59s k jejich deaktivaci. Poté se bude cyklicky opakovat.

```
void poplach (void)
{
  P3DIR |= 0x04; //nastaveni P3 jako vystup
  P3OUT |= 0x04; //blokace zapnout
  P2DIR |= 0x10; //zacni s poplachem

  if (general_action)
  { //je vyhlasen poplach

    if (time_alarm == clk_s) //kazdou celou minutu upravime pocitadlo
    {
      alarm_now++;
    }
    if ((alarm_now % 2) == 0) //kazdou druhou (sudou) minutu poplach
    {

      alarm_on ();
    }
    else //kazdou lichou nehoukame
    {
      alarm_off ();


    }

  }

}
```

Funkce `void poplach(void)` využívá k časování poplachů hodin reálného času, v okamžiku hlášení si uloží aktuální stav počítadla sekund snížený o 1 a počká, až bude stav hodin stejný jako uložený - tedy cyklus probíhal 59sekund. Potom inkrementuje proměnnou `alarm_now`, což způsobí změnu její dělitelnosti dvěma. Pokud lze dělit dvěma beze zbytku, alarm vyvoláme, pokud ne, tak je vypnutý. Ukládání času má na starosti funkce `void test_DATA (void)`.

5.2.7 Test stavu vozidla

Ve funkci `void test_DATA (void)` je volána funkce `void void zamknuto_test (void)` . Tato funkce testuje, zda je vozidlo uzamčeno, a pokud ne, tak ověřuje identitu uživatele pomocí bloku autorizace voláním funkce `void test_autorizace (void)` 5.2.8.

```
void zamknuto_test (void)
{
  if (P2IN & 0x01)
  {
    if (zamknuto == 0)
    { //predtim bylo auto odemknute, ted ho zamykam
      //je potraba zaridit, aby novy poplach
      //po novem uzamceni nebyl ovlivnen predchozimi akcemi
      alarm_now = 2; //promenna, ktera slouzi pro casovani alarmu
    }
    zamknuto = 1; //priznaky
    autorizace_OK = 0;
    pocet_odpoctu = 0;
  }
  else
  {
    // if (zamknuto==1){ //bylo zamknuto a~ted se odemklo
    zamknuto = 0;
    if (!autorizace_OK) //v pripade, ze je otevreno dele,
    //nema cenu neustale shanet kod
    {
      test_autorizace (); //testujeme
    }
  }
}
```

5.2.8 Autorizace

Autorizace uživatele probíhá pomocí klávesnice. Chtěl jsem využít technologii RFID, ale nepovedlo se mi úspěšně komunikovat s čtečkou AXA200 [4]. Proto jsem zvolil jako alternativu klávesnici pro zadání čtyřmístného kódu.

Práce s klávesnicí

Obsluha klávesnice s uspořádáním 4×4 je řešena známou metodou - 4 signály pro sloupce, 4 signály pro řádky. Nastavím logickou jedničku do sloupce a přečtu řádky. Stisknuté tlačítko propojí sloupec a řádek a na příslušném řádku detekuji logickou jedničku. Graficky je tento princip naznačen na obrázku 5.5.

Samotná obsluha klávesnice je řešena funkcí `char klavesnice (void)`, která vrací stisknutou klávesu. Pro kompletnost řešení vrací klávesu X v případě, že není stiknuto žádné tlačítko. Klávesa X není na klávesnici obsažena.

Část kódu funkce `char klavesnice (void)`:

```
char klavesnice (void)
{

  P4DIR |= 0x33; // nastav P4.0,1,4,5 na vystup
  P4OUT &= 0x00;    //neposilej nic na P4
  P4OUT |= 0x01;    //aktivuj prvni sloupec ABCD

  while (buffer <= 3)
  {
    P4OUT &= 0x00;   //neposilej nic na P4
    P4OUT |= 0x01;   //aktivuj prvni sloupec ABCD

    if (P4IN & 0x08)
    {
      buffer_update (); //upraveni pocitadla
      return ('C'); //navratova hodnota
    }
    if (P4IN & 0x04)
    {
      buffer_update ();
      return ('A');
    }
    if (P3IN & 0x40)
    {

      buffer_update ();
      return ('B');
    }
    if (P3IN & 0x80)
    {
      buffer_update ();
      return ('D');
    }

    P4OUT &= 0xFE; //zruseni nastaveni prvniho sloupce
    P4OUT |= 0x02; //nastaveni druheho sloupce 147*

    ...

    P4OUT &= 0xFD; //zruseni nastaveni druheho sloupce
    P4OUT |= 0x10; //treti sloupec 2580

    ...

    P4OUT &= 0xEF; //zruseni nastaveni tretiho sloupce
    P4OUT |= 0x20; //ctvrty sloupec 369#

    ....
```

```
    return ('X');  //nic nestisknuto
  }
}
```

Z ukázky je dobře vidět princip - nastavení logických úrovní řádků, zrušení předchozích. Test na stisknutou klávesu, návrat kódu. Funkce `void buffer_update(void)` upravuje index v poli s ukládaným heslem a přidává prodlevu kvůli neúmyslnému opakovanému stisku klávesy.

```
void (buffer_update) (void)
{
  Delay (20000); //prodleva, pomaha to opakovanym stiskum
  if (buffer < 3) //delka kodu jsou 4 znaky
  {
    buffer++;
  }
  else
  {
    buffer = 0;
  }
}
```

Funkce `char klavesnice (void)` je volána funkcí `void cti_klavesnici (void)`.

```
void cti_klavesnici (void)
{

  char pomocna = klavesnice (); //nacteny znak z klavesnice
  if (pomocna != 'X') //je neco nacteno <
  {
    if (pomocna == '*') //kdyby to byla hvezdicka
    { //nove zadani
      buffer = 3; //nastavime buffer na posledni pozici,
      //cteni znaku o jednu posune na zacatek
    }

    if (pomocna == pole[buffer - 1]) //test na duplicitu stisku
    {
      buffer--;
    }
    else
    {
      pole[buffer] = pomocna; //ukladame data
    }
  }
}
```

Funkce `void cti_klavesnici (void)` zpracuje znaky posílané z klávesnice. Umí detekovat dvojité stisknutí (z bezpečnostních důvodů jsem implementoval tak, aby nebylo možné používat kódy s dvojicí shodných znaků) a také na základě stisknutí hvězdičky umožní zadávat heslo znovu. Uložené heslo zpracuje funkce `int identifikuj (void)`.

```
int identifikuj (void)
{
  if (strcmp4 (pole, user1)) //proste porovnani hesla a~vlozene sekvence
  {
    *pole = "0000"; //nulovani, aby pri opakovanych
        //spustenich nedochazelo k pouziti minule zadaneho hesla

    return (USER1); //vracime kod uivatele
  }
  ...
  else
    return (INVALID_AUTHORIZATION); //prihlaseni se nepodarilo,
    //autorizce neplatna

}
```

Funkce `int identifikuj (void)` vrací kód autorizace. Systém umí zpracovat tři různé uživatele. Tato funkce je společně s funkcí `void cti_klavesnici (void)` volána hlavní funkcí bloku autorizace `void test_autorizace (void)`.

```
 \label{test}
void test_autorizace (void)
{
  int save_error_code; //pomocna promenna

  cti_klavesnici (); //najde
  save_error_code = identifikuj (); //najit kdo se prihlasuje

  if (save_error_code == USER1 || save_error_code == USER2
     || save_error_code == USER3)
  {
    autorizace_OK = 1;
    general_action = 0;
    error_code = save_error_code;
    P2OUT &= 0x00; //prestat houkat
    save_to_ram (); //ulozit kdo se prihlasil
    save_error_code = 0;
    P3OUT &= 0xFB; //blokace vypnout
  }
  else
  {
```

```
    autorizace_OK = 0;
    odpocet ();

  }

}
```

Principem je testování hesla s využitím předchozích funkcí. V případě neúspěšné autorizace je zahájen odpočet doby potřebné na zadání přístupového kódu.

Čas na zadáni kódu

Uživatel musí mít po otevření vozidla k dispozici nějakou dobu na zadání přístupového kódu. Když dobu nevyužije, dojde k vyhlášení poplachu. Po dobu, kterou má uživatel k dispozici systém vydává krátká zvuková znamení.

Tuto problematiku řeší funkce `void odpocet (void)`.

```
void odpocet (void)
{
  if (!odpocet_nastaven) //zrovna neodmerujeme cas
  {
    odpocet_time = ((clk_s + 2)); //ulozime cas
    if (odpocet_time >= 59)
    {
      odpocet_time = 0; //uprava kvuli preteceni minuty
    }
    odpocet_nastaven = 1; // a~odpocitavame
  }
  if (odpocet_time >= clk_s) //kdyz mame jeste cas a~neni aktivni poplach, tak nepipne
  {
    if (!general_action)
    {
      alarm_off (); //alarm vypnout
    }
  }
  else
  {
    alarm_on (); //zapipat
    odpocet_nastaven = 0; //a znovu se bude nastavovat
    pocet_odpoctu++; //scitame
  }

  if (pocet_odpoctu > PRODLEVA) //kdyz to trva dlouho, tak poplach.
  {
    general_action = 1;
    error_code = INVALID_AUTHORIZATION;
    save_to_ram ();
  }
```

```
}
```

Činnost funkce je založena na odměřování času a počtu znamení, která vydá uživateli vozu. Nejprve se uloží čas navýšený o nějakou časovou prodlevu. Poté se počká, až se aktuální čas bude rovnat času uloženému. Přitom se krátce spustí siréna, upraví se nastavení příznaku nastavení odpočtu a stav počítadla odpočtu. Dojde-li k překročení stanového počtu signálů hodnotou `PRODLEVA`, je vyhlášen poplach.

5.2.9 Administrace systému

Administrace systému probíhá pomocí terminálového spojení s následujícím parametry:

- Rychlost: 460800 baudů
- Bitů: 8
- Stop bit: 1
- Parita: žádná
- Řízení toku dat: žádné

Veškerá nastavění jsou dostupná v administračním módu po zadání hesla, data z paměti a nápovědu lze číst bez hesla. Bez zadání hesla nejsou příkazy pro nastavení systémových voleb funkční, pouze vypíší informaci o nutnosti vstupu do administračního módu. Potřebné informace ke konfiguraci získá uživatel zadáním příkazu `HELP`, který vypíše dostupné příkazy s popisem funkce. Do administračního módu lze vstoupit zadáním `SETUP ********` kde hvězdičky představují osmimístné heslo. Administrační mód představuje prostor pro rošíření.

V systému lze měnit následující:

1. Datum
2. Čas
3. Povolit automatickou změnu na letní čas a zpět
4. Heslo pro administraci

Nastavování, data, času, a letního času

Systém je vybaven softwarovými hodinami reálného času. Popis viz. kapitola 5.1.3. Provedl jsem určitá výchozí nastavení popsaná v tabulce 5.2.9.

datum	1.ledna 2008
čas	00:00:00
letní čas	povolen
den v měsíci	úterý

Tabulka 5.6: Implicitní nastavení reálného času

Nastavení času se provede zadáním příkazu `TIME HH:MM:SS`, kde `HH` jsou hodiny `MM` jsou minuty a `SS` sekundy. Podobné je nastavení data zadáním `DATE YY:MM:DD`, kde `YY`představuje rok - pouze poslední dvojčíslí, `MM` měsíc a `DD` den.

Příkazem `DST YES/NO` se provede aktivace nebo deaktivace letního času. V řípadě povolení je ještě třeba zadat aktuální den pro konkrétní výpočet, bez jeho zadání sice dojde k přestavění času, ale může se tak stát o několik dní napřed nebo později oproti správnému datu. Pro zadání dne slouží příkaz `DAY X`, kde `X` je číslo dne. Pondělí je 1, úterý 2 atd.

Realizace tohoto bloku je přímo spojena s detekcí příkazu. Jedná se jen o nastavení hodnot proměnných, nic co by musela realizovat další funkce. Nastavované proměnné jsou `clk_yy`,`clk_mm`,`clk_dd`,`clk_h`,`clk_m`,`clk_s`,`dst` a `day`.

Heslo pro administraci

V administračním režimu je možné změnit heslo příkazem `PASS ********`, kde hvězdičky představují osmimístné heslo. Realizováno nastavením proměnné `password`.

5.3 Model

Po dohodě s vedoucím jsem se rozhodl část systému namodelovat. K vytvoření modelu jsem použil nepájivé pole, samozřejmě FITkit, vodiče, rezistory, tranzistory, spínače, integrované obvody, spínače a několik LED, aby bylo zřejmé, co se v obvodu děje. Seznam součástek je uveden v kapitole 5.3.1 .

5.3.1 Seznam součástek

Rezistory

Označení	Počet
2k2	9
1k	3
100R	6
300	3
10k	1
120R	1

Tabulka 5.7: Použité tranzistory

Tranzistory

Použil jsem 6 bipolárních tranzistorů BC639 z nabídky GM Electronic. Tranzistory jsou typu NPN.

Označení	Počet	Kč/kus
BC639	7	1,50

Tabulka 5.8: Použité tranzistory

Integrované obvody

Označení	Počet	Kč/kus
CMOS 4011	2	1,50

Tabulka 5.9: Použité tranzistory

Spínače

K nastavování logických úrovní v okolí mikrokontroléru jsem použil 6 kusů jednopólových páčkových přepínačů P-KNX1 z nabídky GM Electronic.

Označení	Počet	Kč/kus
P-KNX-1	6	7

Tabulka 5.10: Použité spínače

LED

Pro průběžnou signalizaci stavův nejrůznějších místech obvodů jsem použil LED. Ty jsem později odstranil, aby zbytečně nezvyšovaly příkon zařízení.

Typ	Počet	Kč/kus
5mm červená	10	2
5mm zelená	10	2

Tabulka 5.11: Použité LED

Siréna

Použil jsem piezo bzučák PEB 30 P z nabídky GES Electronics. Za zapůjčení děkuji vedoucímu práce. Zapůjčená varianta se napájí stejnosměrným napětím v rozsahu 2÷5V. Bzučák vydává tón o kmitočtu 2,8kHZ, tento kmitočet je realizován technicky v bzučáku.

5.3.2 Příkon

Příkon periferií

Příkon systému je velmi důležitý. Musí být co nejmenší a s tím ohledem jsem systém navrhoval i realizoval. Měřením jsem zjistil, že hardware realizovaný na nepájivém poli odebírá v klidu, tedy v době, kdy není hlášen poplach skutečně jen velmi malý proud. Naměřil jsem několik jednotek mA. Při měření finální verze se proud ustálil na hodnotě 2,5mA. Při spuštění poplachu potom proud narůstá hlavně vlivem spínačů na kostru, které jsou řešeny pomocí bipolárních tranzistorů. Jeden sepnutý spínač způsobí nárust proudu o cca 22mA.

Označení	Počet	Kč/kus
PEB 15 P	1	38,90

Tabulka 5.12: Použitá siréna

Příkon mikrokontroléru

Mikrokontrolér MSP430F168 instalovaný na FITkitu patří mezi nízkopříkonové. V aktivním režimu, tedy v módu, kdy jsou veškeré periferie k dispozii , dosahuje při kmitočtu 1MHz a napájení 3.3V proud hodnoty nižší než než $400\mu A$.

MSP430F168 nabízí krom aktivního ještě několik úsporných režimů, které se liší množstvím vypnutých periferií. Implementovaný systém využívá časovač odvozený od ACLK, tedy je třeba, aby ACLK zůstal aktivní. Podle [7] by tedy bylo možné procesor uvést až do stavu, kdy odebírá proud o velikosti nižší než $10\mu A$.

Já jsem ale úsporný režim neimplementoval a nakonec jej odstranil i z návrhu. Důvodem je velmi nízká spotřeba už v základním módu. Úspora, kterou bych získal, by sice byla nemalá, avšak v konečném důsledku by z hlediska celého systému nepřinesla nic významného. Zpracoval jsem tabulku 5.3.2, kde uvádím orientační výpočty doby provozu systému.

režim	MSP430	periferie	celkem	doba[h]	doba[den]	doba[rok]
aktivní	$400\mu A$	$2,5mA$	$0,0029A$	17241	718	1,97
úsporný	$4\mu A$	$2,5mA$	$0,002504A$	19968	832	2,28

Tabulka 5.13: Orientační výpočty doby výdrže akumulátoru

Neměřil jsem příkon dynamické paměti, ale výrobce uvádí [9] pro refresh proud maximálně jednotky mA.

Zhodnocení

Jak je patrné z tabulky, systém má velmi nízkou spotřebu. Stává se tak jen doplňkem ke spotřebě automobilu, která je tvořena nápajením paměti rádia a udržováním soustavy řídících jednotek v úsporném režimu. Jedná se o jednotky mA. Změřil jsem i svoje auto. Z obvodů, které představují významnější odběr elektrické energie obsahuje centrální zamykání s dálkovým ovládáním a rádio s pamětí. Odběr celého vozu kolísá (zřejmě nepřesností měřícího zařízení) mezi 2,8mA a 3,2mA. Takový odběr mám v praxi vyzkoušený jako bezproblémový, přitom často i déle než měsíc auto nepoužiji. Z toho usuzuji, že navržený a realizovaný systém je nízkopříkonový a ani v kombinaci s ostatním odběrem v řádu jednotek miliAmpér nemůže působit potíže.

5.3.3 Fyzická realizace modelu

Připojení k PC

Ze všeho nejdříve jsem musel FITkit připojit ke svému notebooku. Z toho jsem měl obavy, protože používám operační systém Microsoft Windows Vista. Moje obavy se potvrdily, nepodařilo se mi nainstalovat potřebné ovladače. Takže jsem nainstaloval další operační systém Microsoft Windows XP. Tam už vše proběhlo podle očekávání správně, tedy tak, jak uvádí web FITkitu [12].

PIN	Funkce	PIN	Funkce
1	Reset	2	Ground
3	Data 7	4	Data 8
5	Data 6	6	Data 9
7	Data 5	8	Data 10
9	Data 4	10	Data 11
11	Data 3	12	Data 12
13	Data 2	14	Data 13
15	Data 1	16	Data 14
17	Data 0	18	Data 15
19	Ground	20	Key
21	DMARQ	22	Ground
23	DIOW-	24	Ground
25	DIOR-	26	Ground
27	IORDY	28	CSEL
29	DMARK-	30	Ground
31	INTRQ	32	IOCS16-
33	DA1	34	PDIAG-
35	DA0	36	DA2
37	CS1FX-	38	CS3FX-
39	DASP-	40	Ground

Tabulka 5.14: Popis IDE konektoru

Datové spoje

Dále jsem musel připravit vodiče pro komunikaci mikrokontroléru s okolím. Na konektor JP9 na desce FITkitu jsem připojil konektor z běžného 80-ti žilového IDE kabelu. Na jeho piny jsem napájel vodiče pro spojení s nepájivým polem, na kterém jsem prováděl simulaci. Vodiče jsem zapojil podle dokumentace [1].

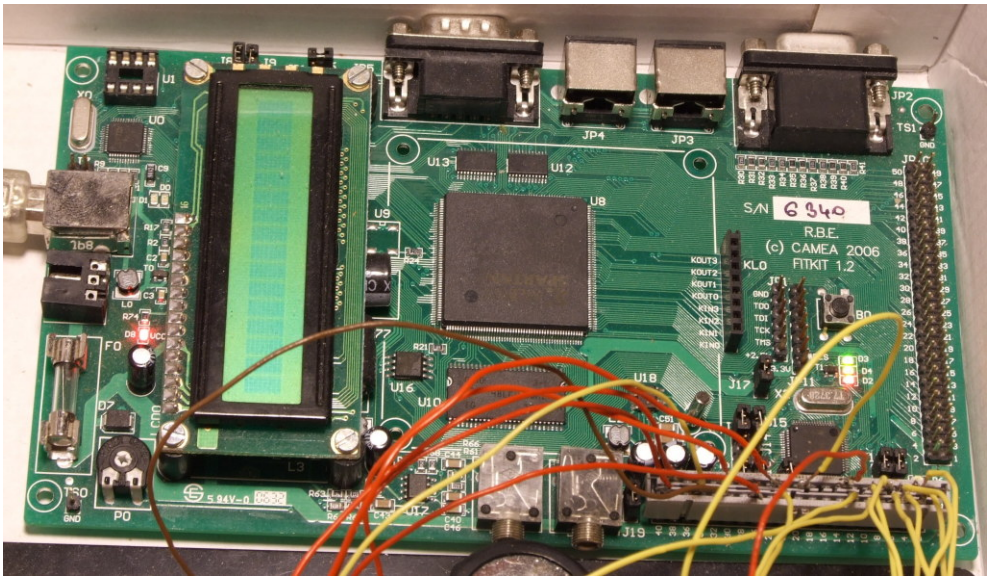
Konektor JP9 a IDE kabel

Nejprve jsem chtěl použít IDE kabel tak jak je a jen do jeho pinů zapojit vodiče. Bohužel to nebylo možné, protože 80-ti pinový kabel ASUS, který jsem měl doma, nevyhovoval z těchto důvodů:

- Zámek proti nesprávné orientaci při montáži
- Zaslepený PIN číslo 20
- Spojené signálové země GND

Zatímco první dva problémy jsou snadno mechanicky odstranitelné, spojené země představovaly větší problém. Popis IDE konektoru uvádím v tabulce 5.1. Převzal jsem ji z [14], kontrolu jsem prováděl i s popisem na webu [11].

Země na pinech 2, 19, 22, 24, 26, 30, 40 byly v konektoru všechny spojeny. To neko-responduje s osazením konektoru JP9 na desce FITkitu [1]. Proto jsem konektor rozebral,



Obrázek 5.9: FITkit s demontovanou klávesnicí a osazeným konektorem

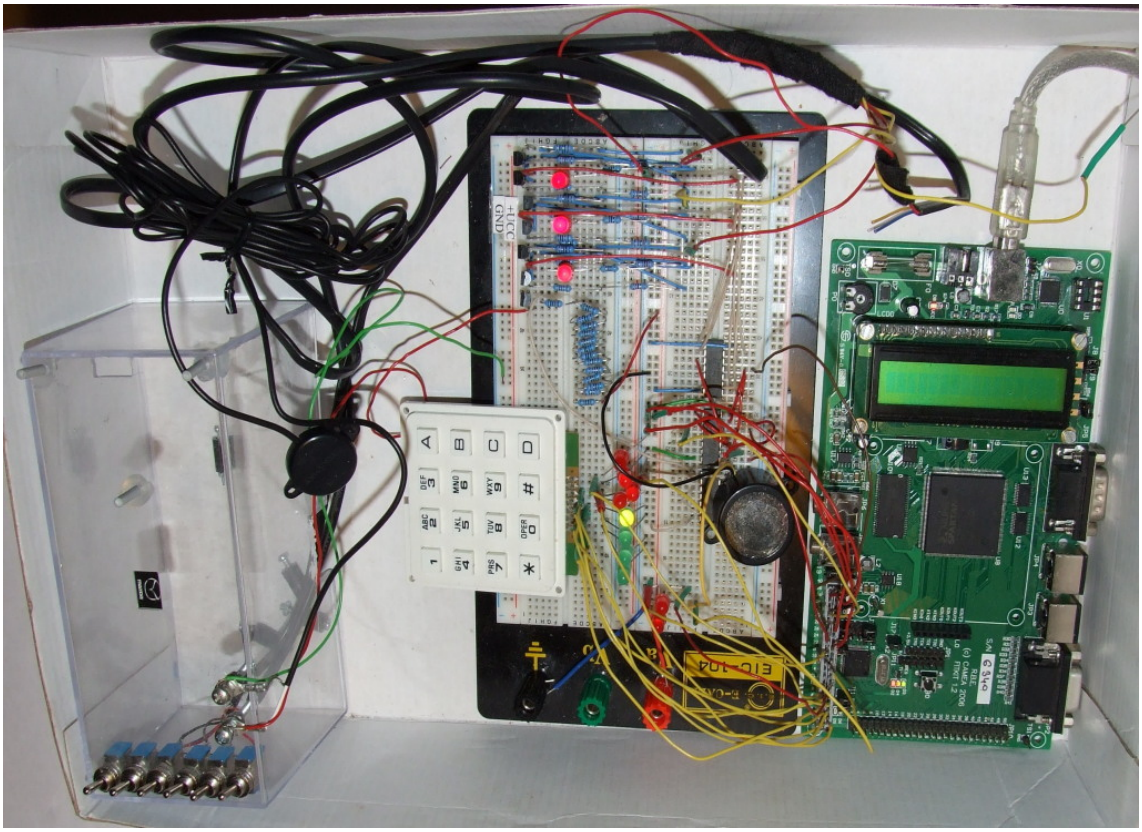
propojení zemí odstranil a na jednotlivé piny, které jsem se rozhodl využít, napájel vodiče. Ukázka provedení je vidět na obrázku 5.9.

Jednotlivé piny jsou umístěny velmi blízko u sebe. Pokud je konektor sestavený, jsou zaříznutý do plochého kabelu, který je zároveň fixuje tak, že se nemohou pohnout. Tento kabel jsem ale odstranil kvůli demontáži propojení zemí. Jeho znovunainstalování jsem zavrhnul jako nereálné, protože je vše velmi malé a nedovedu zaručit korektní sestavení. Obával jsem se zkratu, který by pravděpodobně mohl vést i k poškození FITkitu. Z toho důvodu jsem konektory mírně pilníkem upravil tak, aby byli užší a nemohly se snadno dotknout.

Provedení celého modelu je vidět na obrázku 5.10. Celek jsem položil do papírové krabice, aby jej bylo možné snadno přemisťovat bez rizika poškození nebo odpojení vodičů. V levé části je umístěna plastová krabička s přepínači, uprostřed je osazené nepájivé pole, a v pravé části FITkit.

Obrázek 5.11 ukazuje osazenou desku nepájivého pole připojenou k FITkitu. Dobře je vidět klávesnice, bzučák, pomocné signalizační LED a sada tranzistorů s rezistory.

Zatímco zápis a programování jsem nastudoval z informací a zdrojových kódu dostupných na stránkách FITkitu [12], vstup jsem musel studovat jinde. Výborně mi posloužila internetová stránka Introduction to Programming the MSP430 [16].



Obrázek 5.10: Model systému

PIN	Funkce	PIN	Funkce
1	Nevyužito	2	Nevyužito
3	Klávesnice P4.4	4	Klávesnice P4.5
5	Klávesnice P4.2	6	Klávesnice P4.3
7	Klávesnice P4.0	8	Klávesnice P4.1
9	Klávesnice P3.6	10	Klávesnice P3.7
11	Nevyužito	12	Nevyužito
13	Blokace P3.2	14	`SP_ADR_1` P3.3
15	Nevyužito	16	Nevyužito
17	Nevyužito	18	Nevyužito
19	Alarm P2.4	20	Nevyužito
21	Nevyužito	22	Nevyužito
23	Nevyužito	24	Nevyužito
25	Nevyužito	26	Nevyužito
27	Nevyužito	28	`SP_ADR_2` P1.5
29	Nevyužito	30	Nevyužito
31	`SP_ADR_2` P6.6	32	`SP_ADR_4`
33	Nevyužito	34	Nevyužito
35	Nevyužito	36	`DATA`
37	Nevyužito	38	Nevyužito
39	+5V VCC	40	Ground

Tabulka 5.15: Zapojení konektoru JP9

5.4 Instalace do vozidla

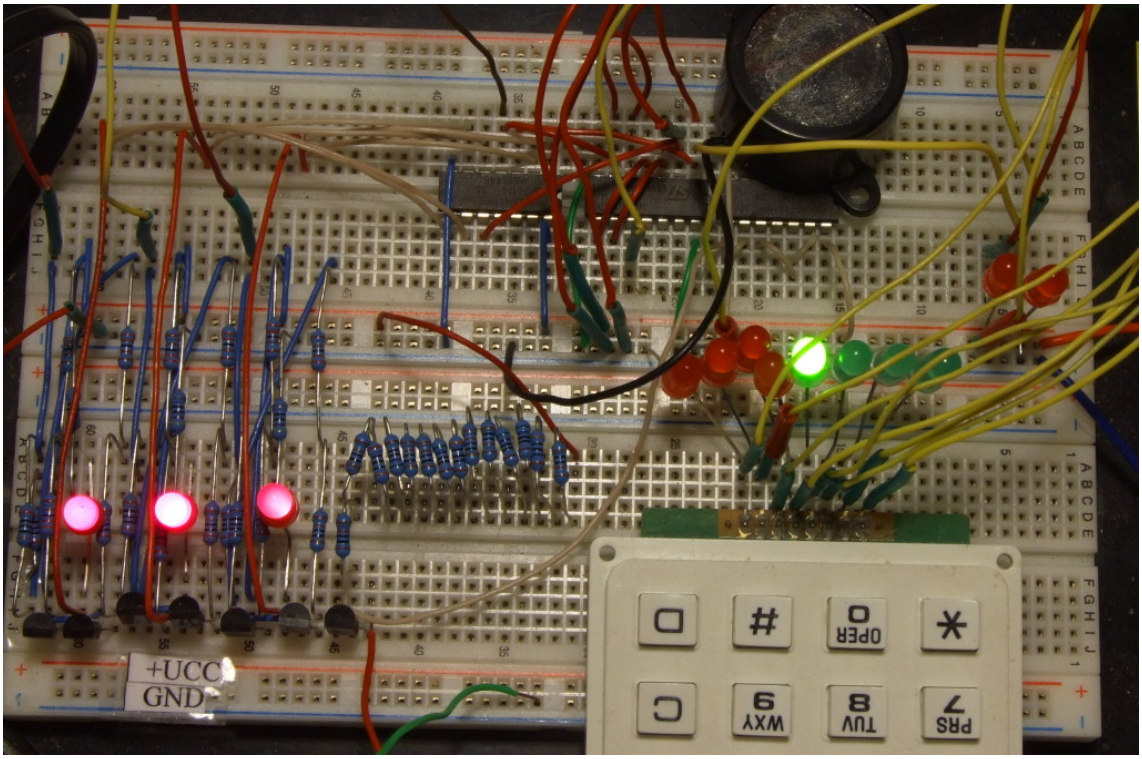
Instalaci do vozidla popisuji ve vztahu k mému autu Škoda Favorit, které dobře znám a mohu tak provést návrhy. Instalace do vizidel jiných typů se může více či méně odlišovat.

5.4.1 Umístění zařízení

Polohou systému ve voze jsem začal. Je zřejmé, že zařízení musí být umístěno tak, aby byly splněny minimálně následující podmínky:

- **Ochrana před škodlivými vlivy** je zřejmé, že nesmíme dopustit, aby na elektrické zařízení tekla nebo odstříkávala voda. Zrovna tak nesmí dojít k poškození chemikáliemi.
- **Nesnadná demontáž** je potřeba jako ochrana systému před jeho vyřazením z provozu útočníkem.
- **Možnost připojení k elektrické síti vozidla** tak, aby bylo dostupné napájecí napětí. Lze jej sice přivést pomocí vodičů téměř kamkoli, avšak tím vzrůstá nebezpečí nalezení a přerušení vedení útočníkem.

Z výše uvedených důvodů plyne, že musíme systém schovat uvnitř vozidla. Za předpokladu jeho bezvadného stavu tam nehrozí vlhkost a ani chemikálie (např. benzín, olej či kyselina z baterie) se za normálních okolností do vozu nedostanou.



Obrázek 5.11: Zapojení na nepájivém poli

Napájení je potom k dispozici téměř všude, avšak nejlepší dostupnost je u přívodu od akumulátoru. Ten je přiveden do pojistkové skříně, odkud je pak rozvedena elektroinstalace po celém autě. Umístění hned u zdroje se mi jeví jako nejlepší. Pojistková skříň se ve Favoritu nachází pod palubní deskou na straně spolujezdce. Je snadno přístupná tak, aby bylo možné případně vyměnit prasklou pojistku, montáž zařízení přímo k pojistkové skříni by nebyla vhodná s ohledem na zabezpečení.

Proto navrhuji demontovat vrchní část palubní desky a ukrýt řídicí jednotku dovnitř. Výhodou toho bude kromě splnění výše zmíněných požadavků i to, že nebude esteticky narušovat interiér a také nebude překážet, protože volného prostoru je uvnitř dost. Dále je běžný Favorit vybavený dveřními spínači, které při otevření dveří spínají na kostru. Standardně slouží k ovládání vnitřního osvětlení vozu, kdy se při otevření rozsvítí a při uzavření zhasne stropní lampička. Provedení spínače je vyobrazeno i s demonstrací jeho funkce na obrázku 5.12. Stisknutý spínač představuje uzavřené dveře, spínač nestisknutý představuje dveře otevřené. Plechové tělo spínače je připevněno ke karoserii, na které je ukostřený minus pól elektrické soustavy vozidla. Při spojení kontaktů je pak ukostřený i vodič připojený ke spínači.

Pro hlídání zadních dveří, kufru a kapoty bude nutné doplnit další spínače, protože z továrny jsou osazeny jen na předních dveřích. Navrhované umístění jednotky ve voze a běžné umístění dveřního spínače názorně ukazuje obrázek 5.13.

Obrázek 5.12: Spínač a demonstrace jeho funkce

5.4.2 Blokování

Z návrhu systému plyne, že je potřeba něco zablokovat. Moje auto je ještě staršího provedení s karburátorem, není v něm příliš mnoho věcí napájených ze sítě nutných k tomu, aby auto jelo. Máme zde pouze následující:

- Spouštěč
- Zapalování
- Elektroinstalace

Pro pochopení dále popsaného principu blokování je nutné znát alespoň přibližně systém, jakým se auto startuje. Předpokládám, že je všeobecně známý, proto jej nebudu rozvádět v tomto textu, nicméně uvádím jej v příloze 8.1.

Na svorku spínací skříňky označovanou číslem 30 je přiveden plus vývod z baterie. Ten se po otočení klíčkem do polohy 1 - zapnuté zapalování - spojí se svorkou 15. Svorka 15 je tedy spínané plus. Tímto spínaným napětím je napájena i zapalovací soustava. V poloze klíčku číslo 2 je aktivován spouštěč, napětí je přivedeno na svorku číslo 50.

Je tedy patrné, že můžeme snadno přerušit spínané vodiče. Bude dobré provést přerušení opět někde v nesnadno přístupném prostoru uvnitř přístrojové desky, kudy vodiče vedou. Rozhodně by nebylo dobré vytvářet rozpojení hned u spínací skříňky, její vodiče jsou přístupné a pro zloděje by nebyl problém blokaci objevit a odstranit.

Přerušíme-li svorku spínaných napětí č. 15, bude bez napájení indukční cívka, která vyrábí jiskru a auto prostě nikdo nenastartuje. Zároveň tento vodič ale přivádí napětí i do pojistkové skříně a k dalším spotřebičům. Tento přívod je vhodné ponechat funkční. Pokud bych přerušil i ten, bylo by auto úplně bez proudu, naprosto mrtvé a nešlo by vůbec nic. To by bylo krajně podezřelé a mohlo by zloději dost napovědět a pomoci s odstraněním. Nesmíme zapomínat, že zloděj auto, které krade, zná.

Další možností je odstavení spouštěče. Já bych ji ale nerealizoval hned z několika důvodů. Tím prvním a nejdůležitějším je opět nápověda zloději. A další věc je ta, že prostě spouštěč není nezbytně nutný k tomu, aby se uvedl motor do chodu. Když se bude spouštěč točit, je to dobře slyšet a může to připomínat závadu. Prostě to nechytne. Při troše štěstí půjde zloděj ukrást jiné auto, takové, které startuje normálně. Se zprovozněním auta, kde motor nasával směs a nezapálil ji, bude mít totiž dost práce. Do válců bude nasáto příliš mnoho

Obrázek 5.13: Umístění jednotky (červená elipsa) a dveřního spínače (modrá elipsa)

směsi a budou díky tomu mokré svíčky, které nedokážou do vysušení zapálit směs. Bylo by tak třeba svíčky vyjmout a nechat uschnout.

Schéma elektroinstalace vozu, zapalovací soustavy a další podrobnosti jsem studoval z dílenské příručky k Favoritu [2].

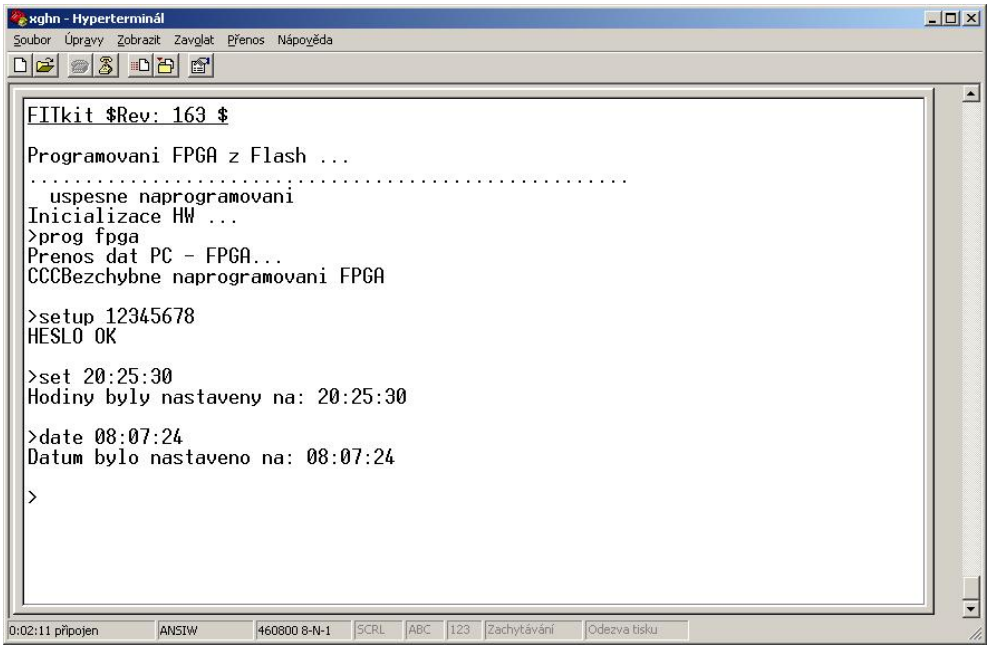
Kapitola 6

Provoz modelu

Nedílnou součástí realizace je i zkušební provoz systému. Při testování jsem vytvořil několik záznamů, kterými se budu zabývat v této kapitole.

6.1 Administrace systému

Při prvním startu systému je třeba nakonfigurovat FPGA. K tomu slouží příkaz `prog fpga` . Dále je potřeba nastavit datum a čas. To je možné ovšem až po zadání hesla pomocí příkazu `setup 12345678` kde 12345678 představuje osmimístné heslo. Při jeho korektním zadání se zobrazí hlášení `HESLO OK`. Poté lze pohodlně konfigurovat systém. Příkazy `SET` a `DATE` nastaví čas a datum. Vše přehledně ukazuje snímek obrazovky terminálu 6.1.

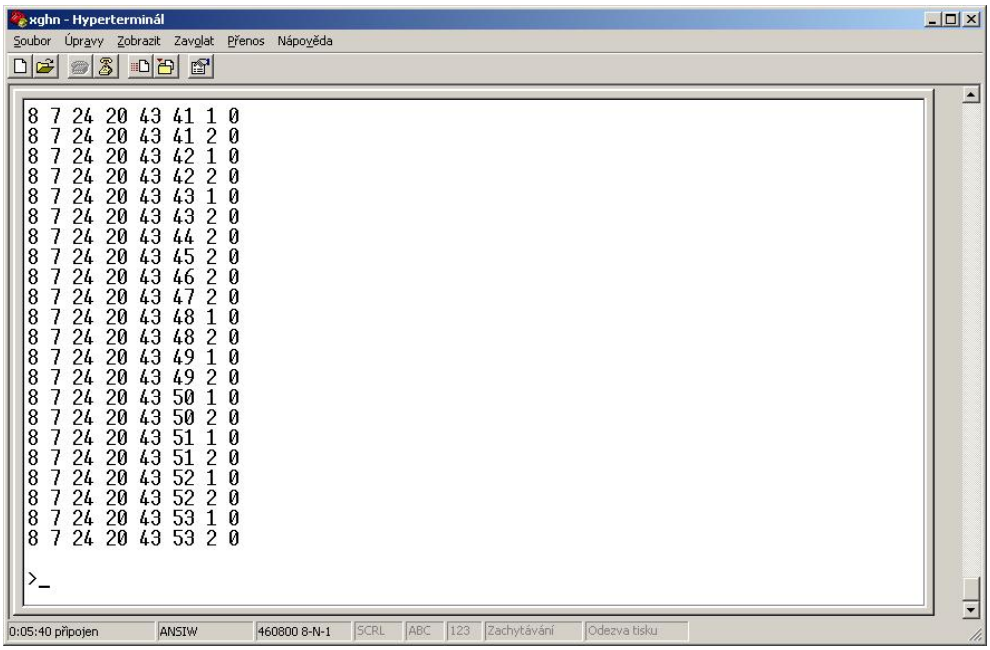


Obrázek 6.1: Nastavení

6.2 Uživatelská data

Systém archivuje velké množství dat. Tato data mohou sloužit například pro diagnostiku systému, ale i pro přehled o bezpečí vozidla. Po jejich vyhodnocení může uživatel, nebo spíše servisní technik, který bude mít systém ve vozidle na starosti, najít vadný spínač nebo také upozornit na časté poplachy, které mohou souviset i s bezpečností lokality, kde vozidlo parkuje.

Uživatelská data jsou zaznamenávána do SDRAM umístěné na FITkitu. Uživateli jsou zasílána prostřednictvím sériového rozhraní na obrazovku terminálu. Pro přečtení paměti RAM slouží uživateli příkaz `READ DATA`. Tento příkaz je dostupný i bez zadání hesla, protože pouze čte a žádným způsobem neovlivňuje nastavení ani činnost systému. Data jsou vypisování ve formátu popsaném v části 5.2.3. Příklad výpisu dat zobrazuje snímek obrazovky terminálu 6.2.



Obrázek 6.2: Uložená data

Na screenshotu 6.2 je velmi dobře vidět, co se dělo:

- K poplachu došlo 24.7.2008
- Poplach trval delší dobu, skončil ve 20:43:53
- Poplach byl vícenásobný, aktivní byly kódy 1 a 2

Nahlédnutím do tabulky chybových kódů 5.2.3 zjistí uživatel, že poplach byl vyvolán otevřením předních dveří.

Kapitola 7

Závěr

Úkolem této bakalářské práce bylo navržení systému pro zabezpečení automobilu. Snažil jsem se tedy nastudovat maximum informací o způsobech zabezpečení automobilů i o možnostech jak je odcizit.

V úvodu se krátce zabývám příčinami nutnosti zabezpečení a možnými riziky. Už znalost těchto rizik může posloužit jako primitivní zabezpečení. Pokud totiž zloději neukážeme, že má co ukrást, tak to hledat nebude, i proto můžeme vidět na parkovištích před supermarkety cedule s textem *Vaše auto není trezor!*.

Dále jsou diskutovány možnosti jak se chránit s ohledem na zmíněná rizika. Tím nejzajímavějším je však návrh systému. Snažil jsem se o maximálně přehledný způsob návrhu s využitím bloků. Kapitola realizace pak poskytuje přehled o tom, co a jak bylo provedeno v oblasti hardware i software.

Mým cílem při volbě tématu bakalářské práce bylo nalezení takové oblasti, kde uplatním a hlavně rozšířím své poznatky z oblasti automobilů. S rostoucími nároky jejich uživatelů na komfort, jednoduchost ovládání a přepravu vzrůstá nutnost vyrábět automobily chytřejší. Už nestačí, že auto jede. Uživatel nechce řadit, nechce složitě parkovat, nechce ale také, aby mu auto někdo ukradl. Chce naopak co nejpohodlněji a co nejbezpečněji cestovat bez zbytečných rizik, komplikací a potíží. Právě zde je podle mých dosud získaných znalostí obrovský prostor pro informační technologie a elektroniku.

Oblast zabezpečení vozidel, kterou jsem se zabýval já, nabízí možnost uplatnění nejrůznějších technologií napříč celým spektrem IT. Můj systém není složitý, ale mohl by být rozšířen o celou řadu zajímavých věcí. Například detekce otisku prstů nebo tváře uživatele, případně i s porovnáním s databází hledaných osob. Samozřejmostí by při takových vlastnostech byla síťová komunikace, dostatečně zabezpečená tak, aby zloděj s notebookem na kapotě nepodstrčil bezpečnostnímu systému falešné údaje.

Možná by se mohla projevit i umělá inteligence. Asi by nebylo špatné vědět o útoku dříve, než nastane, sledovat okolí vozidla kamerami a podezřelé chování vyhodnotit dříve než dojde k vloupání. Jako příklad si představuji situaci, kdy se neautorizovaná osoba pokusí vzít za kliku u dveří auta, kdy se pokusí použít nějaký nástroj v zámku auta. Pokud se vyhlásí poplach v tomto okamžiku, je větší šance, že k vloupání nedojde a pachatel uteče. Systém by však měl být inteligentní, nesmí jako útok vyhodnotit otření kabátu osoby nastupující do vedlejšího auta na parkovišti.

Přínosem této práce je prohloubení znalostí o rizicích automobilu, o elektronických systémech zabezpečení a v neposlední řadě také o jejich návrhu a tvorbě. Náhodný čtenář si také pravděpodobně rozšíří rozhled v problematice zabezpečení a zauvažuje, co by mohl pro svůj vůz udělat.

Kapitola 8

Příloha

8.1 Startování auta

Zde uvádím velmi zjednodušený popis startování a nutných předpokladů pro nastartování. Mou snahou nebylo vytvořit vědecký popis činnosti motoru, ale krátký snadno pochopitelný text.

Zážehový motor (tedy motor poháněný benzínem, nikoli naftou) v autě musí mít pro nastartování dvě základní věci:

- palivo
- jiskru

Jiskru dodává vysokonapěťová zapalovací soustava. Není přitom důležité jak pracuje. Důležitý je ale fakt, že jiskra je nezbytně nutná pro chod motoru. Právě jiskra totiž zapálí palivo.

Palivo musí být ovšem v místě, kam může jiskra. Tedy uvnitř motoru, ve válci. Jak se tam dostane? Musí se nasát (u moderních aut vstříknout) ze sacího potrubí směs připravená karburátorem. Nasaje se ale jen tehdy, existuje-li něco, co saje. Jak známo, saje podtlak. Ten vznikne v motoru tehdy, když píst ve válci koná pohyb dolů a je otevřený sací ventil. Pro nás je důležitý ten pohyb. Pohyb pístu za jízdy způsobuje otáčivý pohyb přenášený na převodovku a kola, písty jsou tedy příčinou pohybu. Při startu ale musíme písty nejprve nějak rozhýbat. K tomu slouží spouštěč. Jedná se o elektromotor, který pomocí ozubeného převodu otočí klikovým hřídelem motoru a způsobí tak pohyb pístů. Tento jev umožní nasátí výbušné směsi do válců, její stlačení a zapálení.

Druhá varianta startu nepotřebuje spouštěč. Je podstatně méně používaná a spočívá v roztočení motoru jiným způsobem. Může se využít polohové energie vozu na kopci či na svahu, může dojít k roztlačení lidskou silou a také k roztažení dalším automobilem. Princip je ale vždy stejný - automobil se uvede do pohybu se stojícím motorem. Když mají kola dostatek otáček, zařadí se vhodný rychlostní stupeň a motor se roztočí od kol. Toho lze využít například při závadě startéru.

Startování bez použití spouštěče nelze použít vždy. U vozidla Favorit s karburátorem, jaké mám já, to není problém, protože neobsahuje katalyzátor a motor je vybaven rozvodovým řetězem. Mnoho jiných aut má ale rozvodový řemen, který při tomto způsobu startu může přeskočit, případně prasknout a zničit tak motor. Téměř všechna vozidla mají katalyzátor, který se takovým startem ničí. Je tedy třeba číst návod k obsluze konkrétního vozu, kde výrobce na podobné riziko, existuje-li, upozorňuje.

Kapitola 9

Seznam příloh

1. 1 kus CD s elektronickou verzí tohoto dokumentu, zdrojovými kódy tohoto dokumentu pro jeho opětovné vygenerování, a sofwarem diskutovaným v této práci
2. textová příloha 8.1 o startování vozu

Literatura

[1] Otto Fucik a Radomir Bardas. Fitkit - input/output interface. http://merlin.fit.vutbr.cz/FITkit/docs/pdfs/hw_io.pdf. Online.

[2] Jaroslav Andrt. *Dílenská příručka Škoda Favorit 115. 135. 136.* AZNP Mladá Boleslav, 1989. I. vydání.

[3] Ing. Jaroslav Budínský. *Polovodičové paměti a jejich použití.* SNTL, 1977. I. vydání.

[4] Elatec. Multireader plug&play board aircoil rs232. http://www.elatecworld.com/fileadmin/user_upload/datasheets_new/reader/125khz/DB_MultiReader.pdf. Online.

[5] GES Electronics. Světelné závory. http://www.ges.cz/?page=index&or=sort&ipp=12&lang=cz&cur=CZK&of=1&gcat=X3_KH&inc=browse. Online.

[6] Česká pojišťovna. Havarijní pojištění. http://www.ceska-pojistovna.net/hav.htm. Online.

[7] Texas Instruments. Msp430x15x, msp430x16x, msp430x161x mixed signal microcontroller (rev. e). http://focus.ti.com/lit/ds/symlink/msp430f168.pdf. Online.

[8] Jablotron. Piezoelektrická siréna. http://www.jablotron.cz/autotechnika.php?pid=products/sa-105. Online.

[9] INC. Micron Semiconductor Products. Synchronous dram. http://merlin.fit.vutbr.cz/FITkit/doc/hardware/datasheet/64msdram.pdf. Online.

[10] Josef Strandel. Fitkit - hodiny reálného času pomocí časovače. https://www.fit.vutbr.cz/study/courses/INC/private/presentations/fitkit.pdf. Online.

[11] WWW Stránky. Ata / ide interface pinout. http://pinouts.ru/HD/AtaInternal_pinout.shtml. Online.

[12] WWW stránky. Fitkit. http://merlin.fit.vutbr.cz/FITkit/uvod.html . Online.

[13] WWW stránky. Fitkit - sdram. http://merlin.fit.vutbr.cz/FITkit/docs/aplikace/app_sdramdemo.html. Online.

[14] WWW Stránky. Ide (ata) bus.
http://www.interfacebus.com/Design_Connector_IDE.html. Online.

[15] WWW stránky. Přestupný rok.
http://cs.wikipedia.org/wiki/Gregori%C3%A1nsk%C3%BD_kalend%C3%A1%%C5%99
. Online.

[16] Adrian Valenzuela. Introduction to programming the msp430.
http://cnx.org/content/m12145/latest/ . Online.

[17] Ministerstvo vnitra odbor bezpečnostní politiky. Krádeže vozidel.
http://www.mvcr.cz/rs_atlantic/project/article.php?id=24388. Online.

[18] otevřená encyklopedie Wikipedie. Letní čas.
http://cs.wikipedia.org/wiki/Letn%C3%AD_%C4%8Das. Online.