VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA STROJNÍHO INŽENÝRSTVÍ
FACULTY OF MECHANICAL ENGINEERING

ÚSTAV VÝROBNÍCH STROJŮ, SYSTÉMŮ A ROBOTIKY
INSTITUTE OF PRODUCTION MACHINES, SYSTEMS AND ROBOTICS

MOŽNOSTI PROPOJENÍ REÁLNÉHO ŘÍDÍCÍHO SYSTÉMU DO PROSTŘEDÍ VIRTUÁLNÍ REALITY

CONNECTIVITY CONTROL SYSTEM TO VIRTUAL REALITY SYSTEM

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — **Jiří Holešovský**
AUTHOR

VEDOUCÍ PRÁCE — **Ing. František Bradáč, Ph.D.**
SUPERVISOR

BRNO 2016

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA STROJNÍHO INŽENÝRSTVÍ

Zadání bakalářské práce

Ústav:	Ústav výrobních strojů, systémů a robotiky
Student:	**Jiří Holešovský**
Studijní program:	Strojírenství
Studijní obor:	Základy strojního inženýrství
Vedoucí práce:	**Ing. František Bradáč, Ph.D.**
Akademický rok:	2015/16

Ředitel ústavu Vám v souladu se zákonem č.111/1998 o vysokých školách a se Studijním a zkušebním řádem VUT v Brně určuje následující téma bakalářské práce:

Možnosti propojení reálného řídícího systému do prostředí virtuální reality

Stručná charakteristika problematiky úkolu:

Proveďte návrh propojení řídících systému PLC a CNC do systému imersní virtuální reality. Pokuste se zpracovat jednoduchý script pro propojení konkrétního systémů s imerzní virtuální reality (IVR) firmy IC.IDO.

Cíle bakalářské práce:

1. Proveďte rozbor možností způsobů získávání provozních dat ze systému SIEMENS SINUMERIC 840D SL.
2. Popište možnosti nalezených způsobů komunikace a komunikačních knihoven. Popište možnosti jejich připojení na programové vybavení IVR IcIDO.
3. Pro vybraný způsob komunikace vytvořte jednoduchý příklad a vytvořte programové moduly pro připojení řídícího systému Sinumeric na prostředí IC.IDO.

Seznam literatury:

OPCFOUNDATION, Unified Architecture. https://opcfoundation.org/about/opc-technologies/opc-ua/, přístup 30.11.2015

SIEMENS, Access MyMachine, http://w3.siemens.com/mcms/mc-systems/en/automation-systems/cnc-sinumerik/cnc-products-functions/sinumerik-integrate/integrate-for-production/pages/access-mymachine.aspx, přístup 30.11.2015

SIEMENS, SINUMERIK 840D sl / 828D Synchronized Actions, https://support.industry.siemens.com/cs/document/109476440/sinumerik-840d-sl-828d-synchronized-actions?dti=0&pnid=14599&lc=en-WW, přístup 30.11.2015

SIEMENS, SINUMERIK Integrate for production, http://w3.siemens.com/mcms/mc-systems/en/automation-systems/cnc-sinumerik/cnc-products-functions/sinumerik-integrate/integrate-for-production/pages/integrate-for-production.aspx, přístup 30.11.2015

ESI-GROUP, IC.IDO, https://www.esi-group.com/software-services/virtual-reality/icido, přístup 30.11.2015

PYTHON, https://www.python.org/, přístup 30.11.2015

Termín odevzdání bakalářské práce je stanoven časovým plánem akademického roku 2015/16

V Brně, dne

L. S.

doc. Ing. Petr Blecha, Ph.D. ředitel	doc. Ing. Jaroslav Katolický, Ph.D. děkan

ABSTRAKT

Tato práce se zabývá možnostmi propojení reálného řídicího systému s prostředím imersní virtuální reality. V první části jsou zmíněny možnosti pohybování modelu v IC.IDO. Dále se reěerže zabývá komerčními způsoby získávání dat z řídicího systému Sinumerik 840D sl a následně naprogramovatelnými způsoby tj. bez nutnosti dokupování dalších licencí nebo softwaru. V experimentální části byly napsány dva g-kódy. Jeden pro posuv stroje v ose x a druhý pro čtení polohy stroje v ose x v závislosti na čase. Pro IC.IDO byl vytvořen skript, který podle naměřených dat z řídicího systému pohybuje modelem v IC.IDO.

ABSTRACT

This thesis deals with options of connecting the real control system to the immersive virtual reality system. Options of moving an object in IC.IDO are mentioned in the first part of the thesis. The theoretical part deals with commercial types of gaining data from control system Sinumerik 840D sl and with programmable types of gaining data which means that there is no need to buy other licenses or software. Two g-codes were made in the experimental part. The first one provides feed of the machine in x axis and the second one reads position of the x axis of the machine and actual times. A script for IC.IDO which moves a model in IC.IDO exactly as the machine moved was written.

KLÍČOVÁ SLOVA

IC.IDO, IDO skript, virtuální realita, Sinumerik 840D sl, synchronní akce, příkaz WRITE, Siemens Integrate for production

KEYWORDS

IC.IDO, IDO script, virtual reality, Sinumerik 840D sl, synchronized actions, WRITE command, Siemens Integrate for production

BIBLIOGRAFICKÁ CITACE

HOLEČOVSKÝ, J. *Možnosti propojení reálného řídicího systému do prostředí virtuální reality*. Brno: Vysoké učení technické v Brně, Fakulta strojního inženýrství, 2016. 44 s. Vedoucí bakalářské práce Ing. František Bradáč, Ph.D.

POD'nKOVÁNÍ

Tímto bych chtnl podnkovat vedoucímu mé práce, panu Františkovi Bradáļovi za cenné rady a vynikající zpnsobu pśi tvośení této práce. Velký dík patśí také Ondśeji Andrśovi a Jiśímu Kováśovi, kteśí byli ochotní mi vysvntlovat dílļí problémy a doplśovat mé znalosti. Nakonec bych chtnl podnkovat svým rodiļm a pśítelkyni za podporu bnhem studií.

ČESTNÉ PROHLÁŠENÍ

Prohlašuji, že tato práce je mým původním dílem, zpracoval jsem ji samostatně pod vedením Ing. Františka Bradáče, Ph.D. a s použitím literatury uvedené v seznamu.

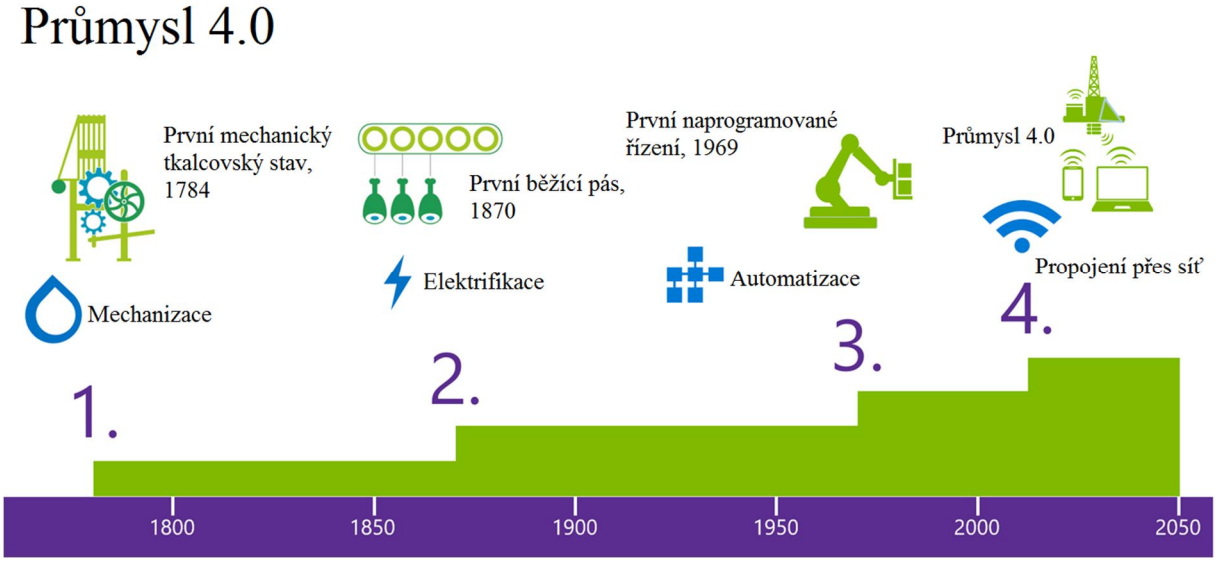
V Brně dne 27.5.2016 ……

Holešovský Jiří

OBSAH

1 ÚVOD

V poslední době se stále častěji setkáváme s automatizací výrobních procesů. Jedná se převážně o nahrazení lidí u opakovaných manuálních prací stroji. Tento proces je nazýván čtvrtou průmyslovou revolucí nebo také Průmyslem 4.0. Proces je úzce spojen s propojováním výrobních strojů mezi sebou a spoluprací autonomních strojů s lidmi (*viz Obr. 1*) [1]. Konkrétně propojení řídicího systému s virtuální realitou, kterým se zabývá tato práce, má širokou škálu použití. Při navázání spojení lze sledovat celý výrobní proces z libovolného místa v prostředí virtuální reality. Při rozšíření tohoto softwaru by bylo možné skutečný stroj ve virtuální realitě ovládat.



Obr. 1) Vývoj průmyslu za posledních 200 let [2]

1.1 Motivace

Na trhu se zvedá poptávka po propojování virtuálního světa s tím skutečným. Je to způsobeno převážně nárůstem používání virtuálních simulací nebo prezentací. Pro velké firmy je důležité, aby se montážní nebo výrobní linky postavily ergonomicky nebo aby se sladily s okolním prostředím [3]. Konstruktéři mohou předcházet chybám buď velmi dobrým plánováním nebo vytvořením úplné simulace všech objektů právě v prostředí imersní virtuální reality. Další využití virtuální reality je při prezentování obráběcích strojů. Zákazník si může stroj nejen prohlédnout ze všech stran, ale může si z něj i virtuálně odmontovat libovolné díly nebo si provést názorný řez strojem [4]. Pro firmy je toto řešení rovněž výhodnější i připravit projekční plátno se senzory a počítačem je snazší, než připravit několik obráběcích center.

1.2 Cíle

Cílem této práce je navrhnout způsoby propojení řídicího systému Sinumerik 840D sl s prostředím virtuální reality IC.IDO. V experimentální části bylo za cíl stanoveno, aby se ve virtuální realitě pohyboval libovolný objekt tak, jako posuv v ose X u tříosé frézky MAS MCV 754 quick *(viz Obr. 2)*.

Obr. 2) Třiosá frézka MAS MCV 754 quick a řídicí systém Sinumerik 840D sl

1.3 Uspořádání práce

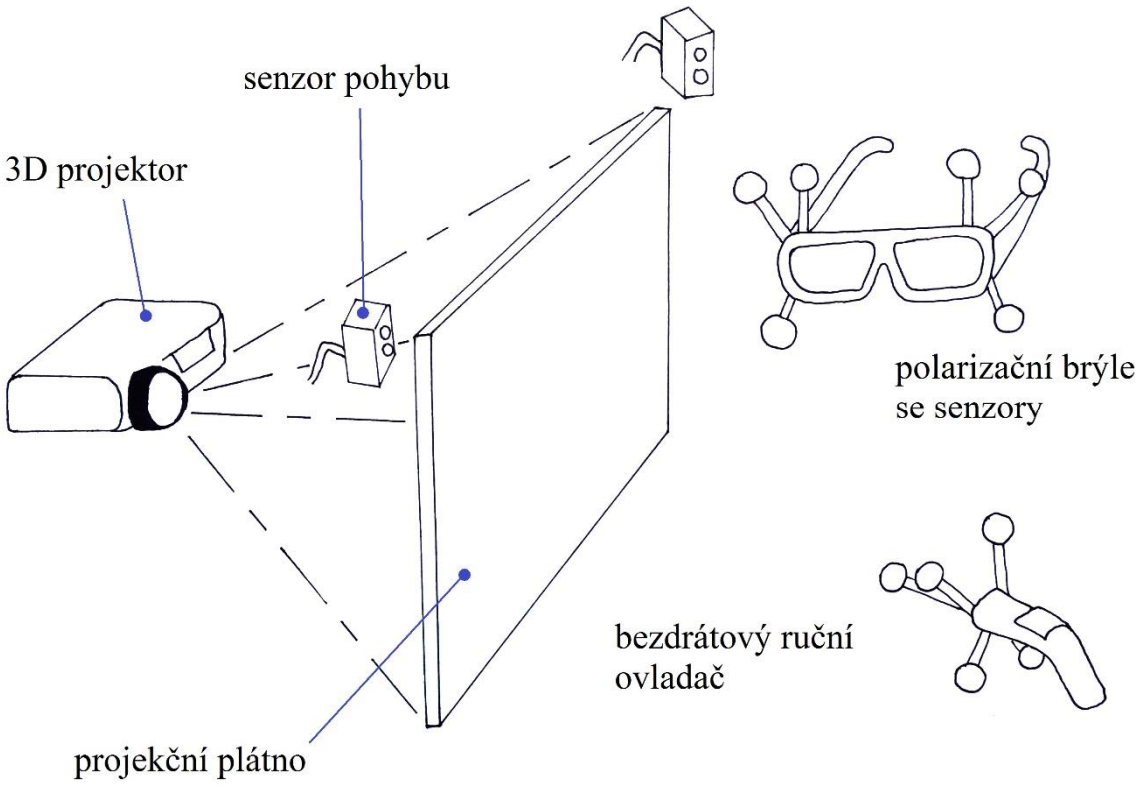
Druhá kapitola se zabývá možnostmi a funkčností IDO skriptů, pomocí kterých jsme schopni ovládat scénu. Následně byla pozornost věnována získávání provozních dat z řídicího systému. Ve třetí kapitole najdeme celou řadu těchto možností, které nabízí přímo výrobce řídicího systému – Siemens. Mimo komerční programy na získávání provozních dat jsou zde zmíněny i metody, které se dají naprogramovat, i když má systém téměř základní výbavu.

2 POHYB OBJEKTU V IC.IDO

Existuje celá řada možností, jak ovládat scénu v prostředí virtuální reality. Hlavním ovládacím prvkem je bezdrátový ruční ovladač, který má i několik klasických funkčních tlačítek. Pohled na scénu je kontrolován tak, že je snímána poloha polarizačních brýlí. V rozích promítacího plátna (tzv. powerwallu) jsou umístěny snímače pohybu, které detekují pohyb bílých kuliček, umístěných na ovladači a brýlích (*viz Obr. 3*). Na každém ovladači jsou minimálně tři snímací kuličky, aby senzory dokázaly detekovat přesnou polohu ovladače v prostoru. Ve výsledku se pohybuje model ruky ve virtuální realitě přesně tak, jako ruka operátora a pohled je ovládán natáčením a pohyby hlavy do stran [5].

Další možnost, jak ovládat pohyb objektů ve scéně je pomocí vstupních dat. Program IC.IDO pomocí naprogramovaného kódu přečte vybraná vstupní data a na základě hodnot provede pohyb objektu ve scéně.

Výše zmíněné programování se provádí pomocí IDO skriptů, kterými se budeme zabývat v následujícím odstavci.



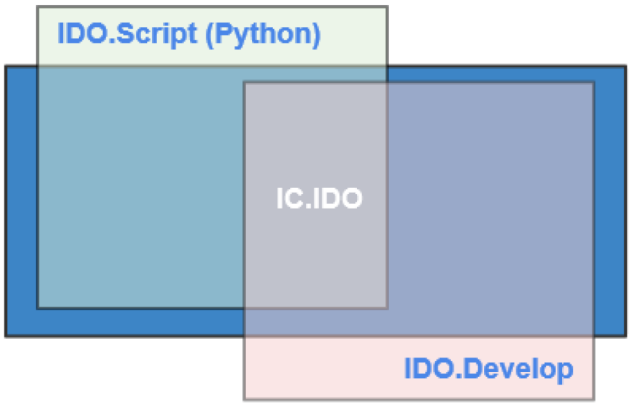
Obr. 3) Hlavní prvky virtuální reality IC.IDO

2.1 IDO skripty

K podrobnému nastavení akcí na scéně slouží IDO skripty. Pomocí IDO skriptů je možné nastavit příslušné externí ovladače nebo definovat pohyby modelů.

Kód pro IDO skript je tvořen v jythonu, což je implementace pythonu v javě (*viz Obr. 4*) [6]. Samotný program může být vytvořen v libovolném textovém editoru

a po přidání koncovky (.py) následně otevřen v uživatelské konzoli IC.IDO. Otevření programu provedeme zvolením nabídky „File“ a volbou „Load script“. Program je aktivován potvrzením pole „attach“. První část programu se provede ihned po načtení skriptu, což může být například definování proměnných a zbylá část programu je provedena až po aktivování pole „attach“. Jakmile program udělá příslušné operace, odpojíme ho zvolením pole „detach“.



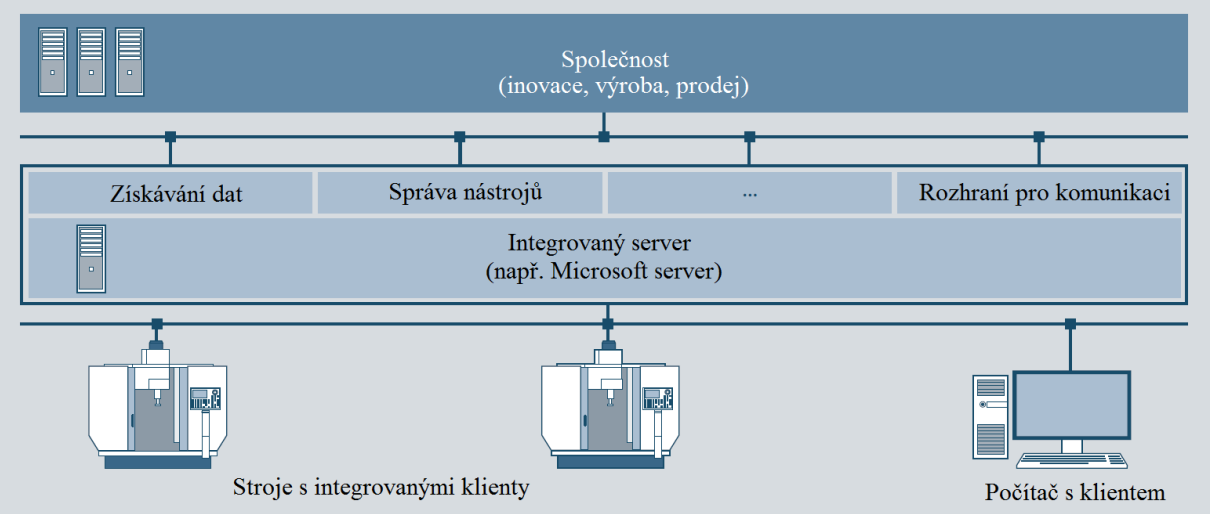
Obr. 4) Schéma architektury systému IC.IDO. (IDO. Develop je vývojové prostředí pro IC.IDO) [7]

3 ČTENÍ DAT Z ŘÍDICÍHO SYSTÉMU

Způsobů, jak získávat provozní data z řídicího systému Sinumerik 840D sl je celá řada. Komerční programy jako například programy ve skupině Integrate for production většinou získávají data, která jsou pro vizualizaci pohybu stroje nedostatečná. Jedná se většinou o stavy nástrojů, chybová hlášení nebo o počty vyrobených kusů [8]. Některé z těchto programů umí číst i informace o poloze nástroje, ale z důvodu vysoké ceny na trhu nebyla příležitost si tyto funkce vyzkoušet. Jelikož je předpokládáno, že se bude vývoj podobných programů dál rozvíjet, budeme se jimi zabývat alespoň v teoretické části. V druhé části jsou popsané tipy a triky, jak provozní data číst i bez zakoupených programů. Jejich výhodou je, že si je můžeme dle požadavků libovolně nastavit. Mezi hlavní nevýhody patří časová náročnost vytvoření kódu a s tím spojené nedostatečné množství jednoduchých návodů nebo příkladů ke psaní. Manuály od firmy Siemens předpokládají značnou míru programátorských znalostí a zkušeností.

3.1 Siemens Integrate for production

Integrate for production je soubor programů od firmy Siemens, který umožňuje propojení výrobního stroje s externími zařízeními (*viz Obr. 5*). Programy mohou komunikovat s počítači i přes internet, s mobilními telefony nebo přímo s dalšími stroji, které jsou součástí výrobní linky. Díky navázání takové komunikace v rámci celého výrobního procesu se zlepší zpětná vazba mezi dílčími výrobními procesy a tím i celková flexibilita produkce [8].



Obr. 5) Schéma připojení výrobních systémů k externím zařízením [8]

3.1.1 Access MyBackup

Při častých pádech nebo výměnách hardwaru je nutné, aby rychle došlo k obnově původních dat. K tomuto účelu slouží rozšíření Access MyBackup, které zprostředkovává odesílání vybraných provozních dat na externí úložiště. Po pádu hardwaru je potom možné ihned nahrát do stroje ztracená data zpět. Především se jedná o informace o stavech nástrojů nebo uložené nastavení. Získávání dat o poloze frézky by pravděpodobně bylo možné, ale jelikož externí úložiště zajišťuje jiná společnost, než Siemens, byla tato možnost zamítnuta [8, 9].

3.1.2 Access MyMachine

V nabídce programů na ovládání řídicího systému na dálku najdeme i Access MyMachine. Komunikace probíhá přes šifrované spojení TLS/SSL po internetu. Veškerou komunikaci mezi strojem a operátorem lze ukládat. Způsob komunikace je rozdělen do tří úrovní: jednoduchý přístup – single access, konference – conferencing a dálkové ovládání – remote step 7. Při jednoduchém přístupu získá uživatel vzdálenou plochu, možnost nahrávat a číst data ze systému a zaznamenávat procesy, které lze později přehrát. Při režimu konference se mohou připojit i další lidé ke sledování procesů. Úroveň remote step 7 je nejpokročilejší ze všech – umožňuje uživateli připojit se přímo k PLC (programovatelný logický automat) a dělat diagnózu nebo řešit problémy s PLC. Výhodou je, že tento modul nemusí být nainstalován přímo na stroji. Pro naše použití by bylo toto programové vybavení ideální [10].

3.1.3 Analyze MyConditions

Program Analyze MyConditions má za cíl zabraňovat prostojům ve výrobě. Program problémům zabraňuje tak, že sleduje vybrané hodnoty a na základě měření předpoví potenciální chybu a může včas přivolat servis. Jsou sledovány základní statusy stroje. Například když se zvýší určitá teplota nad kritickou hodnotu, začne tuto hodnotu sledovat a snaží se předpovědět potenciální problém [8].

Na počátku sledování se stanoví tzv. „machine fingerprint“. Jedná se o soubor běžných provozních parametrů stroje. Program včas upozorní uživatele, když dojde k překročení těchto běžných hodnot. Pro uživatele to znamená, že je nutné zjistit příčinu odchylky a stroj opravit. Kupříkladu, když se zvýší spotřeba energie u posuvu, je možný výskyt nečistot. Pomocí „machine fingerprintu“ lze rovněž porovnávat více strojů mezi sebou [11]. Je očekáváno, že starší stroj bude mít vyšší běžné hodnoty pro spotřebu energie, než stroj nový.

Upozornění je posíláno pomocí SMS nebo šifrovaně přes internet vždy na konci PLC cyklu. Program nabízí velké množství možností, jak a kam upozornění odesílat.

Výhodou je jednoduchá instalace – není potřeba, aby byly přidávány další programy do systému, ale stačí pouze aktivovat klienta, který už je v ovládacím panelu. Další výhodou je jednoduchost prohlížení naměřených údajů, jelikož se provádí v internetovém prohlížeči [11]. Pro naše použití, sledování osy X, tento program není příliš vhodný. Dokáže sice odesílat jakákoliv CNC data a stavy stroje přes internet, což by usnadnilo i čtení dat ve vizualizačním programu IC.IDO, ale z důvodu odesílání dat až na konci PLC cyklu, vhodný není.

3.1.4 Manage MyPrograms

Pokud je požadována rychlá distribuce a dobrá správa CNC programů ve výrobě, je vhodné zvolit rozšíření Manage MyPrograms. Program zajišťuje, že vytvořená výrobní data jsou k dispozici po celé továrně [8]. Proces potom probíhá následovně: konstruktér vymodeluje CAD/CAM díl, připojí k němu výkres a odešle vše do výroby. Operátor si v řídicím systému otevře náhledové okno a vyhodnotí data jestli dříve, než se převedou do dalších dat řídicího systému. Pomocí programu Manage MyPrograms může ihned konstruktérovi odeslat zpětnou vazbu. Vše je integrováno v ovládacím panelu. Pro náš účel není tento program příliš vhodný, jelikož zajišťuje pouze sdílení dat potřebných k výrobě [12].

3.1.5 Manage MyTools

Výrobní proces je rovněž urychlen, pokud jsou včas dodávány potřebné nástroje. To zprostředkuje program Manage MyTools [8]. Na řídicím panelu jsou do knihovny nástrojů

uloženy všechny informace o nástrojích, jako jsou: délky, zaoblení, tolerance nebo jejich pozice uložení ve skladech. Seznam nástrojů může být doplněn i obrázky. Parametry nástrojů jsou editovány přímo ve stroji. Program běží na panelu a není potřeba další počítač. Manage MyTools dokáže sám organizovat pohyb nástroje po výrobě. Pokud NC program potřebuje určitý nástroj a není aktuálně na skladě, je program schopen ho objednat od výrobce. Mimoto je sledována i životnost nástrojů. Stroj si tak pamatuje data o všech nástrojích a při odložení nástroje na delší dobu, ho není potřeba při novém použití měřit [13]. Jelikož Manage MyTools zajišťuje pouze záležitosti týkající se nástrojů, není pro naše použití dostatečný.

3.2 Příkaz WRITE

Kvůli možné ztrátě dat během produkce, jsou používány příkazy EXTOPEN, WRITE a EXTCLOSE, která dokáží data archivovat [14]. Zápis probíhá tak, že je nejprve nakonfigurováno zařízení, do kterého bude zapisováno a potom je vepsán do výrobního programu příslušný příkaz WRITE. Nejprve se tedy podívejme na konfiguraci zařízení pro ukládání.

Nastavení probíhá v souboru „extdev.ini“. Šablona souboru se nachází v paměti systému v card/siemens/sinumerik/nck. Šablona je nakopírována do card/user/sinumerik/nck [15]. Do konfigurační ho souboru jsou následně vepsány informace o zařízení, do kterého bude ukládáno. Určující parametry jsou uživatelské jméno, heslo, server, název sdílené složky nebo i přesný soubor k zapisování. Po jakékoliv změně souboru extdev.ini je důležité celý stroj resetovat. Zde je příklad nadefinování externího zařízení v souboru „extdev.ini“ [15]:

```
WINDOWS SHARE
=============
/DEV/EXT[1-9] = "//[DOMAIN]USERNAME%PASSWORD@SERVER/SHARE,
/FILE_NAME/[,[O|A]]"
```

Příkaz WRITE musí být použit s příkazem EXTOPEN, který otevře předdefinované zařízení v cílové složce. Samotný příkaz WRITE stanoví, kam přesně a jaká data budou ukládána. Zápis dat proběhne až nakonec programu a je doplněn příkazem EXTCLOSE, který zařízení, na které je zapisováno, uzavře. Celá syntaxe čtení dat je následující [16]:

```
DEF INT <ERROR>
DEF STRING[<N>]<OUTPUT>
…
EXTOPEN(<ERROR>,"<EXTG>",<PROCESSING MODE>,<USAGE MODE>,<WRITE
MODE>)
…
<OUTPUT>="OUTPUT DATA"
WRITE(<ERROR>,"<FILENAME>"/"<EXTG>",<OUTPUT>)
…
EXTCLOSE(<ERROR>,"<EXTG>")
```

Příkaz EXTOPEN má následující parametry [18]:

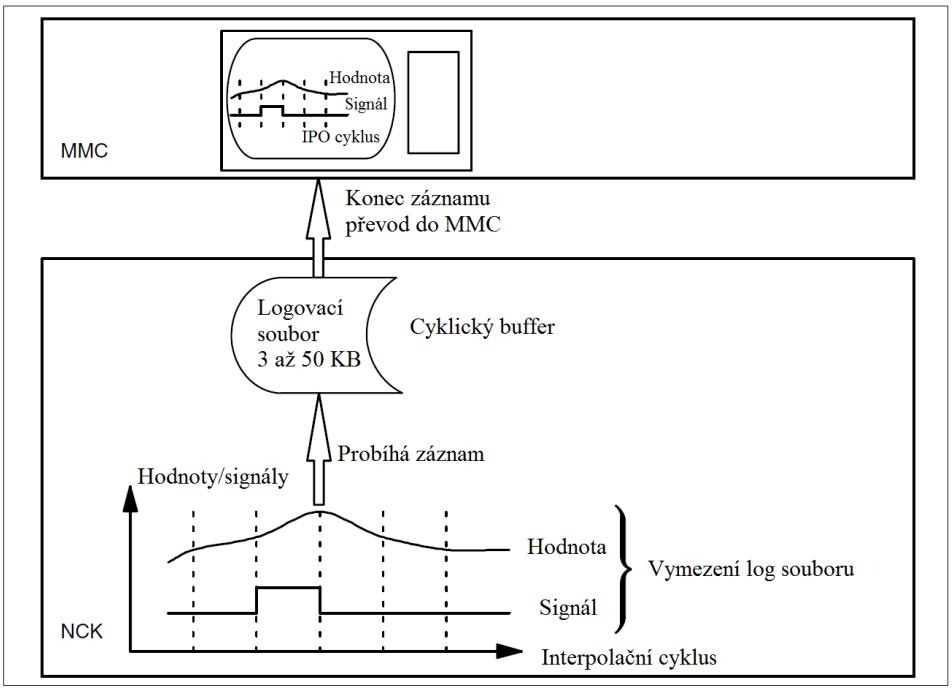
Tab 1) Parametry příkazu EXTOPEN

Příkaz	**Popis**
STRING	Definuje řetězec
EXTOPEN	Příkaz na otevření externího zařízení
<ERROR>	Proměnná na vypsání chyby
Hodnota	*Význam*
0	*Žádná chyba*
1	*Nepovolená ukládací cesta*
2	*Ukládací cesta nenalezena*
3	*Soubor nenalezen*
4	*Neplatný typ souboru*
10	*Plný soubor*
11	*Soubor je používán*
12	*Zdroje nejsou k dispozici*
13	*Nedostatečná oprávnění*
20	*Ostatní chyby*
<EXTG>	Identifikátor externího zařízení
Název zařízení	*Popis zařízení*
"LOCAL_DRIVE"	*Lokální flash paměť, nutné specifikovat cestu uložení: "LOCAL_DRIVE/slozka/soubor.txt"*
"dev/ext/[1-9]"	*Zařízení na síti, nutné specifikovat v souboru "extdev.ini"*
<PROCESSING MODE>	Režim procesu zapisování
Parametr	*Popis*
"SYN"	*Synchronní zápis, program je zastaven, dokud zápis neproběhne*
"ASYN"	*Asynchronní zápis, program běží nezávisle na zapisování*
<USE MODE>	Režim používání zařízení
Parametr	*Popis*
"SHARED"	*Více kanálů má zároveň přístup k zařízení/souboru*
"EXCL"	*Zařízení/soubor je používán pouze v jednom kanálu*
<OUTPUT>	Proměnná pro data k zápisu
OUTPUT DATA	Data k zápisu
WRITE	Příkaz na zapsání dat do souboru
<FILE NAME>	Název souboru, do kterého se zapisují data
EXTCLOSE	Zavře zařízení, do kterého byla data zapsána

3.3 Zaznamenávání aktuálních proměnných

Pokud potřebujeme zaznamenávat aktuální proměnné během běhu interpolačního cyklu, může být zvolena metoda zaznamenávání aktuálních proměnných (*viz Obr. 6*). Vybrané hodnoty

jsou zapisovány do souboru Ălogñ o definované velikosti (3 aģ 50 KB). Jedná se o synchronní akci, tudíģ zápis probíhá v násobcích taktu interpolaļního cyklu. Na poļátku si vybereme promĎnné a zvolíme velikost logovacího souboru. Tento soubor funguje jako cyklický buffer, to znamená, ģe kdyģ se soubor naplní, zaļnou se hodnoty od zaļátku pŚepisovat [19]. Metoda zaznamenávání aktuálních promĎnných je pro naģe pouģití vhodná. Z dŢvodu nedostateļného mnoģství pŚíkladŢ a studijních materiálŢ, nebyla metoda v praktické ļásti práce pouģita.



Obr. 6) ZpŢsob získávání dat pomocí zaznamenávání aktuálních promĎnných. MMC je komunikace mezi obsluhou a strojem (man machine communication), NCK je numerické Śídící jádro (numeric control kernel) [19]

3.4 Synchronní akce

Obecná metoda, jak získávat hodnoty aktuálních promĎnných bĎhem chodu stroje jsou synchronní akce. Synchronní akce bĎģí na pozadí spuģtĎného programu v násobcích taktu interpolaļního cyklu [19]. Jedná se o takto formulované pŚíkazy:

Tab 2) PŚíklad synchronní akce [19]

Komponent	Identifikaļní ļíslo	Frekvence	G kód pro podmínku	Podmínka	Akce	G kód pro akci	Akce
PŚíklad:	IDS=1	EVERY	G70	$AA_IM[X] > 15	DO	G71	POS[X]=100

Tab 3) Parametry synchronní akce [19]

Identifikační číslo	
Neurčeno	Synchronní akce probíhá jen v následujícím bloku, probíhá jen v automatickém módu
ID = od 1 do 255	Probíhá stále, dokud není spuštěna další synchronní akce se stejným identifikačním číslem nebo není zastavena příkazem CANCEL(i) probíhá jen automatickém módu
IDS = od 1 do 255	Probíhá ve všech ovládacích módech
Frekvence	
Žádná	Akce se provádí v každém interpolačním cyklu
WHENEVER	Provádí akci vždy, když je splněna podmínka
FROM	Provádí akci od prvního splnění podmínky
WHEN	Provede akci jednou, pokud je splněna podmínka, dále už podmínku nesleduje
EVERY	Akce se provede jednou, když je splněna podmínka a potom vždy, když je podmínka znovu splněna
Podmínka	
$AA_IM[X] > 15	Pokud je aktuální hodnota X-ové osy v souřadnicovém systému stroje větší, než 15
Akce	
POS[X]=100	Posuv v ose X na hodnotu 100

Pro přenos informací o posunu nástroje u frézky jsou synchronní akce nejlepším řešením. Nejenže umožňují čtení aktuálních proměnných v jednom kanále během spuštěného programu, ale zároveň jsou schopny běžet i po celou dobu zapnutí stroje. Pro stanovené použití nebylo zvoleno programování pomocí synchronních akcí, ale bylo směřováno k využití příkazu WRITE. Důvodem byla nižší úroveň programování a s tím spojené nižší nároky na zkušenosti a znalosti programátora.

4 EXPERIMENTÁLNÍ ČÁST

Jak již bylo v první polovině této práce naznačeno, experimentální část se zabývá navázáním komunikace mezi řídicím systémem Sinumerik 840D sl s prostředím virtuální reality IC.IDO. K navázání komunikace na straně řídicího systému byl zvolen cyklus s příkazem WRITE. Stojí za zmínku, že mimo softwarových řešení zabudovaných přímo v řídicím systému, lze tuto úlohu vyřešit i připojením dalších zařízení nezávislých na řídicím systému. Tím může být například instalace laserového měřiče vzdálenosti, který by sledoval pohyby stroje a odesílal data v reálném čase do prostředí IC.IDO. Toto řešení by bylo výhodnější v tom, že by zařízení běželo nezávisle na řídicím systému, tudíž při chybě na stroji by nedošlo ke ztrátě sledování obráběcího centra virtuální realitou. Nevýhodou je ovšem složitý software pro měření a ve vedlejší řadě výše pořizovacích nákladů na další senzoriku.

4.1 IDO skript

Úkolem vytvořeného IDO skriptu je ovládat pohyb objektu „deskaň ve scéně „scene_object.idoň. Program přečte z předdefinovaného umístění textový soubor, který obsahuje data o pohybu kostky. V souboru je matice, kde v prvním sloupci jsou zaznamenané vteřiny a v druhém sloupci je poloha osy X v milimetrech. Objekt „deskaň se tedy bude posouvat přesně tak, jak se hýbe stroj.

Při otevření skriptu v IDO konzoli proběhne import používaných funkcí, zpracování vstupních dat a definování řízeného objektu. Při potvrzení tlačítka „attach scriptň bude probíhat pohyb tělesa.

4.1.1 Zpracování vstupních dat

Po importu použitých funkcí bylo nutné zpracovat vstupní data. Příkazem `open()` byl externí soubor „X_position.txtň vložen do proměnné a následně byla data příkazem `map()` oddělena mezerami [20]. Proces byl završen výpisem dat na obrazovku.

Zpracování vstupních dat proběhlo následovně:

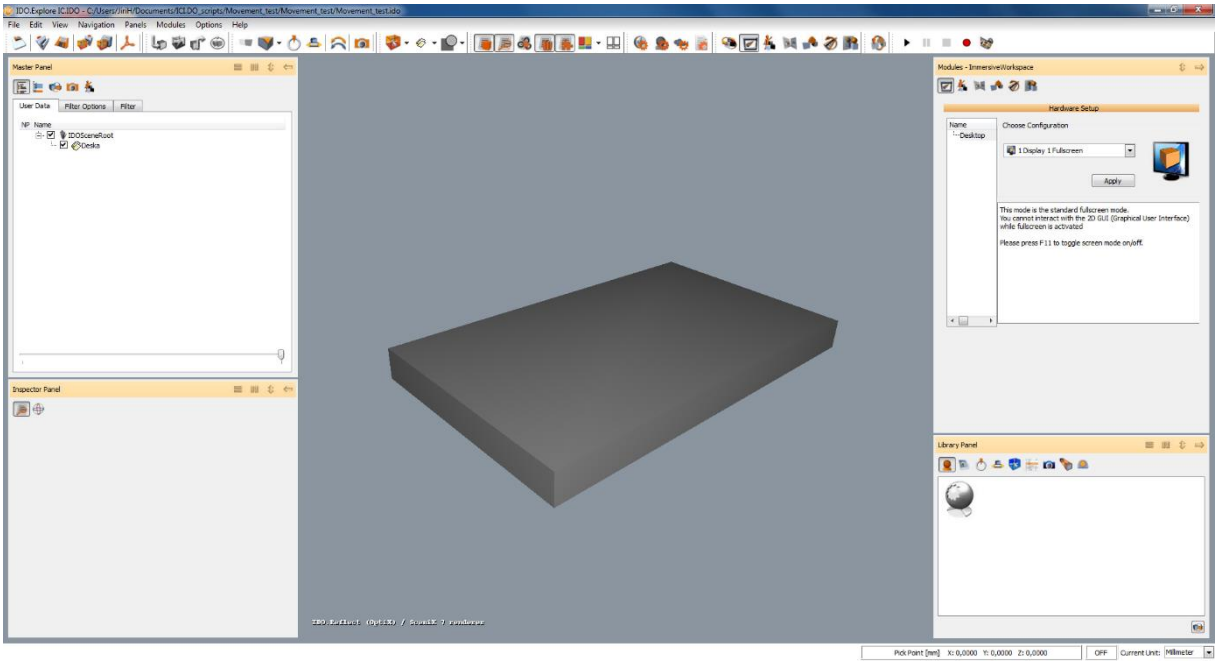
```
f=open ('C:\Users\JiriH\Documents\ICI.DO_scripts\Movement_test\
X_position.txt' , 'r')
g=[map(float,line.split(' ')) for line in f]
print g
```

Tab 4) Hodnoty v souboru X_position.txt pŚed a po zpracování

X_position.txt v intervalu 0 aģ 1 sekunda, pohyb konstantní rychlostí z 0 na 50 milimetrŢ	
Poļáteļní forma	Zpracovaná data
0 0 0.1 5 0.2 10 0.3 15 0.4 20 0.5 25 0.6 30 0.7 35 0.8 40 0.9 45 1 50	[0.0, 0.0], [0.1, 5.0], [0.2, 10.0], [0.3, 15.0], [0.4, 20.0], [0.5, 25.0], [0.6, 30.0], [0.7, 35.0], [0.8, 40.0], [0.9, 45.0], [1.0, 50.0]

4.1.2 Definování tĐesa v IDO skriptu

Objektem je krychle vytvoŚená v Autodesk Inventor, uloģená do formátu programu CATIA. Po vloģení modelu do scény je nutné buŅ definovat nový souŚadnicový systém, nebo pŚi importu potvrdit volbu Āzachovat geometriiñ. PŚíkazy `getSceneObjectListByName()` a `getFirstSceneObject()` byl model Ādeskañ vloģen do promĐnné `sceneObject`. Jeho souŚadnicový systém byl vloģen do promĐnné `coordinateSystemList` pŚíkazem `getAttribute()` [20].



Tab 5) Model Ādeskañ v prostŚedí IC.IDO

Vložení modelu do proměnné `sceneObject` [7]:

```
soList = JC_IDOSceneObjectManager.getSingleton().
getSceneObjectListByName('Deska')
sceneObject = soList.getFirstSceneObject()
```

Uložení souřadnicového systému do `coordinateSystemList` [7]:

```
attributeId = JC_IDOCoordinateSystemList.getAttributeAttributeId()
coordinateSystemList = sceneObject.getAttribute(attributeId)
```

4.1.3 Pohyb tělesa

Po připojení skriptu je zaveden uživatelský souřadnicový systém. Poté je stanoven čas začátku běhu programu. Pro čas byla použita funkce `time.clock()` [7]. Hlavní WHILE cyklus běží od začátku běhu programu po poslední časový údaj v souboru „X_position.txt". Při splnění podmínky, že je čas běhu vyšší nebo roven času ze souboru „X_position.txt", je proveden posun tělesa na souřadnici, která je zaznamenána u příslušného času. Posun je popsán translační maticí `matrixTransJC` a proveden příkazem `setTransformBSbyWS()` [7, 21].

Po připojení programu („def attach") je dvěma FOR cykly vytvořena matice [7]:

```
for i in range(0, 4):
      fData.append([])
      for j in range(0, 4):
            fData[i].append( float( dpData.getDouble( i + j * 4 ) ) )
```

Následně je matice použita jako uživatelem definovaný souřadnicový systém [7]:

```
userDefinedCurrentCoordinate = MatrixF44( fData );
userDefinedCurrentCoordinateInv = userDefinedCurrentCoordinate.
inverse()
```

Před začátkem běhu hlavního WHILE cyklu je uložen počáteční čas do proměnné `cas_zacatku` příkazem:

```
cas_zacatku=time.clock()
```

`Time.clock()` uloží do proměnné `cas_zacatku` hodnotu uplynulých sekund od referenčního data 1. ledna 1980. Celý cyklus je řízen tímto odpočtem [7].

WHILE cyklus proběhne tehdy, když `cas_behu` cyklu je menší, než poslední časový záznam (zde `t_overall`) z „X_position.txt". Na počátku je `cas_behu` roven nule, jeho hodnota se aktualizuje při každém dalším proběhnutím WHILE cyklu [22].

```
while cas_behu < t_overall:
    cas_behu=time.clock()-cas_zacatku
```

V hlavním cyklu je umístěna podmínka, která zajišťuje, aby byl objekt posunut ve správný čas na správnou souřadnici. Podmínka je splněna, pokud je `cas_behu` vyšší nebo roven časovému údaji z „X_position.txt", uloženého v poli „y" [23]. Do proměnné „a" se ukládá vzdálenost, o kterou se má těleso „deska" posunout. Hodnota se dělí tisíci pro převod z milimetrů na metry.

```
if cas_behu >= g[row][0]:
      if cas_behu == 0:
            a=(g[row][1])/1000
      else:
            a=(g[row][1]-g[row-1][1])/1000
```

Dále je urļena translaļní matice s posunutím o Ăaň [7, 26]:

```
matrixTransJC = JC_Matrix_F_4_4()
matrixTransJC.set(1, 0, 0, a; 0, 1, 0, 0; 0, 0, 1, 0; 0, 0, 0, 1)
matrixTrans = Matrix_F_4_4_Conversion.instance.
toExternal(matrixTransJC)
```

V závĐru cyklu jsou pŚíkazy k posuvu tĐlesa `sceneObject` podle transformaļní matice [7]:

```
matrixWS = sceneObject.getTransformWS()
matrixTemp1 = userDefinedCurrentCoordinateInv.times(matrixWS)
matrixTemp2 = matrixTrans.times(matrixTemp1)
matrixTemp3 = userDefinedCurrentCoordinate.times(matrixTemp2)
sceneObject.setTransformBSbyWS(matrixTemp3)
```

Nakonec pŚíkaz `row=row+1` pŚiļte hodnotu jedna k aktuálnímu ļtenému Śádku a program se tak pŚesune v dalģím cyklu v rámci ļtení hodnot na dalģí Śádek.

4.2 G kód

Byly vytvoŚeny dva programy. Jeden má za úkol pohybovat strojem dle potŚeby a druhý mĐŚí aktuální souŚadnici v ose X v závislosti na ubĐhlém ļase. PŚi mĐŚení budou programy spuģtĐny ve dvou kanálech a bude provedeno pŚíslušné nastavení, aby bylo toto spuģtĐní moģné. Jelikoģ pouģívaný Śídicí systém má pouze jeden kanál a ani simulátor od firmy Siemens - SinuTrain spouģtĐní ve více kanálech nepodporuje, bylo rozhodnuto, ģe si experiment vystaļí pouze s funkļními dílļími programy [25].

Prvním programem je ĂMOVE_X.MPFň, který zajišŠuje lineární posuv v ose x, který bude mĐŚen. PŚíkaz `G54` urļí souŚadnicový systém a pŚíkazem `F100` je dána rychlost posuvu na 100 mm/min. Posuv na poģadovanou souŚadnici zajistí pŚíkaz `G1 X500`, zpĐt na poļátek pŚíkaz `G1 X0`. Program uzavírá pŚíkaz `M30` [27].

```
N1 G54
N5 F100
N10 G1 X500
N15 G1 X0
N20 M30
```

4.2.1 MĐŚení polohy

V programu ĂTIMER.MPFň jsou nejprve definovány vģechny promĐnné vļetnĐ doby mĐŚení a uloģeny ļasy spuģtĐní. Cyklus CYKLUS probíhá do té doby, dokud se nenaplní pŚedem urļený ļas mĐŚení. Jako mĐŚítko ļasu je pouģitý ļas od zaļátku epochy, který má rozliģení jedna vteŚina [21]. KdyģubĐhne vteŚina, probĐhne zápis aktuálních dat do matice. Do prvního

sloupce matice `matrix` se zapíše aktuální čas a do druhého sloupce souřadnice X v milimetrech, ze systémové proměnné `$AA_IM[X]` [19].

Při definování proměnných na začátku programu je hlavním parametrem `cas_behu`, který určuje, jak dlouho bude program měřit. V našem případě byla hodnota stanovena na 10 sekund:

```
DEF INT cas_behu=10
```

Následně byly v programu určeny potřebné časové hodnoty, podle kterých bude program pracovat. Systémové hodnoty času `$A_SECOND`, `$A_MINUTE` a `$A_HOUR` byly převedeny na sekundy a vloženy do proměnných. `Start_seconds` a `start_cyklus` mají hodnotu počtu vteřin od začátku epochy a `seconds` a `cas_ted` jsou na počátku nulové [28].

```
start_seconds = $A_SECOND+$A_MINUTE*60+$A_HOUR*3600
start_cyklus=$A_SECOND+$A_MINUTE*60+$A_HOUR*3600
seconds=$A_SECOND+$A_MINUTE*60+$A_HOUR*3600-start_seconds
cas_ted=$A_SECOND+$A_MINUTE*60+$A_HOUR*3600-start_cyklus
```

Hlavní cyklus probíhá, je-li splněna podmínka, že `seconds` je menší, než `cas_behu`. `Cas_behu` je uživatelem stanovená doba měření a `seconds` je časový údaj, který zaznamenává, jak dlouho měřící cyklus už běží. Celý měřící program pak probíhá v tomto cyklu [16]:

```
CYKLUS:
…
IF seconds<cas_behu GOTOB CYKLUS
```

Měření je řízeno jednou IF podmínkou. Pokud je `cas_ted` větší nebo roven `0.5`, proběhne sekvence příkazů, které zaznamenají polohu v ose X a čas od počátku měření do matice `matrix`. `Cas_ted` nabývá hodnot pouze 0 nebo 1. Hodnoty 1 nabyde tehdy, pokud uplyne další vteřina a po vykonání měření se opět vynuluje. Hodnota polohy nástroje na ose X se získává ze systémové proměnné `$AA_IM[X]` [16].

```
IF cas_ted>=0.5
seconds=$A_SECOND+$A_MINUTE*60+$A_HOUR*3600-start_seconds
matrix[row,0]=seconds
matrix[row,1]=$AA_IM[X]
row=row+1
start_cyklus=$A_SECOND+$A_MINUTE*60+$A_HOUR*3600
ENDIF
cas_ted=$A_SECOND+$A_MINUTE*60+$A_HOUR*3600-start_cyklus
```

4.2.2 Výstup nahraných dat

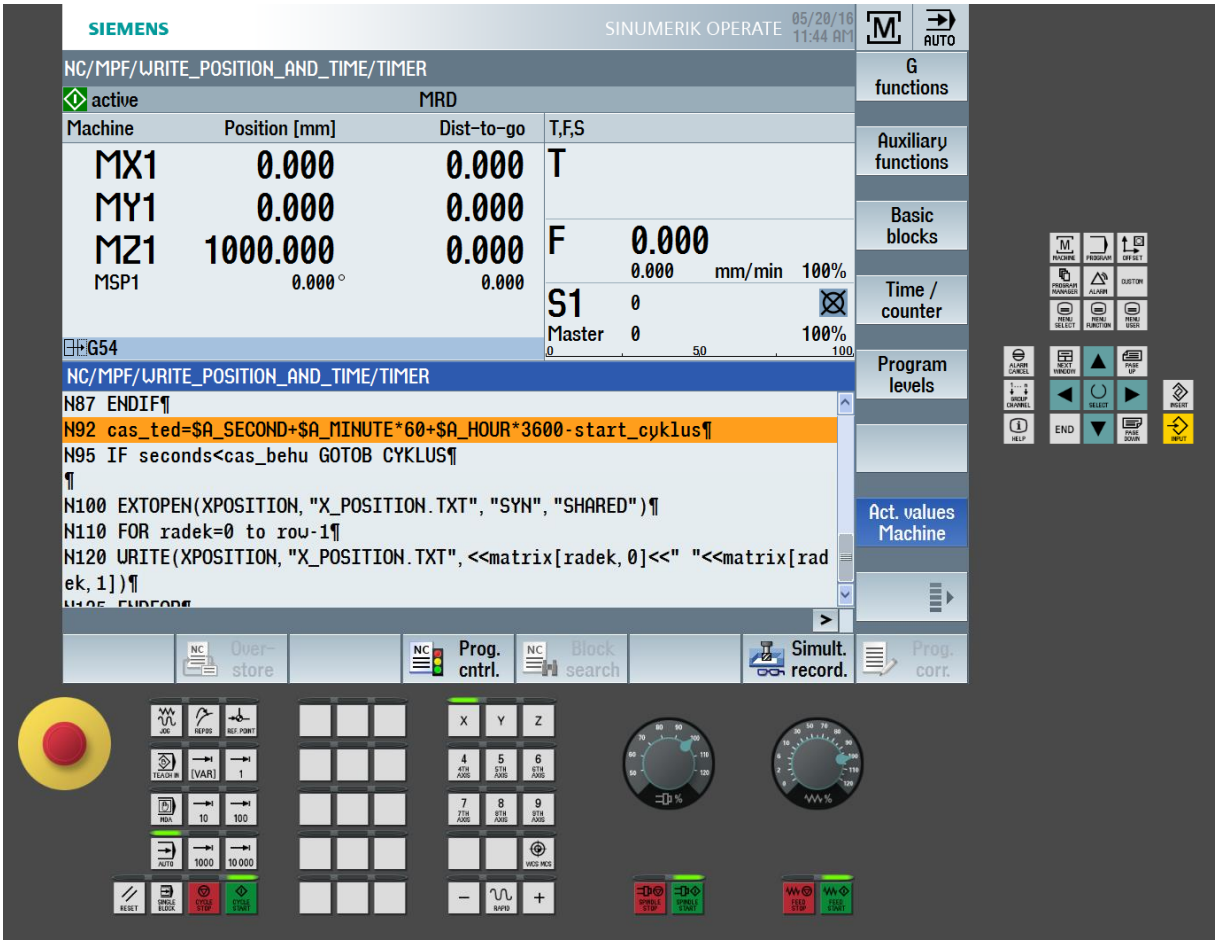
Po proběhnutí cyklu jsou data zapsána do požadovaného souboru, v našem případě „X_POSITION.txt“. Cílová cesta je zvolena příkazem `EXTOPEN()` [14]. FOR cyklus provede zápis všech členů vytvořené matice příkazem `WRITE()` do textového souboru. Data z matice `matrix` jsou převedena konverzí << na formát STRING (řetězec) a mezi hodnotu času a polohy je vložena podobným způsobem mezera [16].

```
EXTOPEN(XPOSITION,"X_POSITION.TXT","SYN","SHARED")
FOR radek=0 to row-1
WRITE(XPOSITION,"X_POSITION.TXT",
<<matrix[radek,0]<<" "<<matrix[radek,1])
ENDFOR
```

Při měření stroje v klidu po dobu 10 sekund (*viz Obr. 7*) vypadá soubor „X_position.txt“ následovně:

```
1 0
2 0
3 0
4 0
5 0
6 0
7 0
8 0
9 0
10 0
```

V prvním sloupci jsou sekundy záznamu a v druhém sloupci poloha stroje v ose X v milimetrech. Soubor „X_Position.txt“ má přesně takový formát, jaký je otevírán v IC.IDO.



Obr. 7) Průběh měření stroje v klidu v programu SinuTrain

4.3 Přenos dat mezi řídicím systémem a IC.IDO

V našem případě se sdílením dat myslí pouze sdílení souboru „X_Position.txt“. Jelikož program běžící v systému Sinumerik odesílá data až po dokončení cyklu, v našem případě je dostačující data přenést po flash disku. Při potřebě přenášet data rychleji by byla data nahrávána po síti na server, odkud by byla čtena IDO skriptem. Nastavení ukládání na server by se provedlo v souboru „extdev.ini“ a v příkazu `WRITE`.

5 SHRNUTÍ

Cíl práce byl navrhnout možnosti propojení řídicího systému s prostředím virtuální reality. Bylo zjištěno, že způsobů je celá řada a záleží především na požadavcích zákazníka, to jest hlavně, jaká data chce získávat. Při zvolení možnosti nákupu softwaru od firmy Siemens bude záležet hlavně na finanční možnostech a aktuální nabídce firmy. Už při tvoření této práce byly některé části z Integrate for production díky své nespolehlivosti staženy z nabídky. Jestliže se zákazník rozhodne programy vytvořit, budou rozhodující hlavně programátorské schopnosti. V práci bylo ukázáno propojení Sinumeriku 840D sl s virtuální realitou pomocí dvou programů běžících ve dvou kanálech na straně řídicího systému a IDO skriptu na straně IC.IDO. Vybraný způsob má sice výhodu ve snazší struktuře kódu, naopak je složitější program spustit. Spouštění ve dvou kanálech je značně komplikované kvůli potřebě měnit základní nastavení stroje. Elegantnější metodou je vytvoření programu pomocí synchronních akcí, které běží v taktu interpolátoru. Nejenže je touto metodou možné sledovat data s mnohem vyšším rozlišením, ale není nutné, aby měl řídicí systém více kanálů. Další výzkum se bude zabývat upravením hotového programu na takový, který běží, jako synchronní akce.

6 SEZNAM POUŽITÝCH ZDROJŮ

[1] MAŘÍK, Vladimír. Průmysl 4.0: Národní Iniciativa [online]. 2015 [cit. 2016-05-20]. Dostupné z: http://www.spcr.cz/images/priloha001-2.pdf

[2] Industrie 4.0 und das Internet der Dinge [online]. [cit. 2016-05-20]. Dostupné z: http://olivia1381.rssing.com/chan-52614519/all_p1.html

[3] Virtuální výroba [online]. [cit. 2016-05-20]. Dostupné z: https://www.esi-group.com/cz/softwarova-reseni/virtualni-vyroba

[4] Virtuální realita [online]. [cit. 2016-05-20]. Dostupné z: https://www.esi-group.com/cz/softwarova-reseni/virtualni-realita

[5] ZDEŇNK, Tůma. Verifikace stavu reálné výrobní linky pomocí virtuální reality. Brno, 2014, 77s. Vysoké učení technické v Brně, Fakulta strojního inženýrství. Vedoucí práce Doc. Dr. Ing. Radek Knoflíček.

[6] Welcome to the Jython Wiki! [online]. [cit. 2016-05-20]. Dostupné z: https://wiki.python.org/jython

[7] *IDO.Scripts: HOWTO - Use IDO.Script*. Jurastrasse 8, 70565 Stuttgart Germany: ESI Software Germany GmbH, 2014.

[8] SINUMERIK Integrate for production: Future-proof IT solutions for machine tools [online]. [cit. 2016-05-20]. Dostupné z: http://w3app.siemens.com/mcms/infocenter/dokumentencenter/mc/Documentsu20Brochures/E20001-A1790-P610-V1-7600.pdf

[9] SINUMERIK Integrate Access MyBackup: Intelligent backup for SINUMERIK 840D sl [online]. [cit. 2016-05-20]. Dostupné z: http://w3app.siemens.com/mcms/infocenter/dokumentencenter/mc/Documentsu20Brochures/SINUMERIK_Integrate_Access_MyBackup_EN.pdf

[10] SINUMERIK Integrate Access MyMachine: Secure remote maintenance with Access MyMachine [online]. 2015. [cit. 2016-05-20]. Dostupné z: http://w3app.siemens.com/mcms/infocenter/dokumentencenter/mc/Documentsu20Brochures/SINUMERIK_Integrate_Access_MyMachine_EN.pdf

[11] SINUMERIK Integrate Analyze MyCondition: Acquire machine states with Analyze MyCondition [online]. 2015. [cit. 2016-05-20]. Dostupné z: http://w3app.siemens.com/mcms/infocenter/dokumentencenter/mc/Documentsu20Brochures/SINUMERIK_Integrate_Analyze_MyCondition_EN.pdf

[12] SINUMERIK Integrate Manage MyPrograms: Efficiently organize and manage NC programs with Manage MyPrograms [online]. 2015 [cit. 2016-05-20]. Dostupné z: http://w3app.siemens.com/mcms/infocenter/dokumentencenter/mc/Documentsu20Brochures/SINUMERIK_Integrate_Manage_MyPrograms_EN.pdf

[13] SINUMERIK Integrate Manage MyTools: Transparently and efficiently manage tools with Manage MyTools [online]. [cit. 2016-05-20]. Dostupné z: http://w3app.siemens.com/mcms/infocenter/dokumentencenter/mc/Documentsu20Brochures/SINUMERIK_Integrate_Manage_MyTools_EN.pdf

[14] Output on an external device [online]. 2014 [cit. 2016-05-20]. Dostupné z: http://www.industry.siemens.com/topics/global/en/cnc4you/tips_and_tricks/pages/output-on-an-external-device.aspx

[15] *SINUMERIK 840D sl: Writing to the hard disk of the PCU 50 from a part program* [online]. 2013 [cit. 2016-05-20]. Dostupné z: https://support.industry.siemens.com/cs/document/79626464/sinumerik-840d-sl%3A-writing-to-the-hard-disk-of-the-pcu-50-from-a-part-program?dti=0&lc=en-US

[16] *Programming Guide 03/2004 Edition* [online]. 2004 [cit. 2016-05-20]. Dostupné z: https://cache.industry.siemens.com/dl/files/744/108681744/att_825253/v1/PGA_0304_en.pdf

[17] *CNC4you: Practical knowledge for the shopfloor* [online]. 2015 [cit. 2016-05-20]. Dostupné z: http://www.industry.siemens.com/topics/global/en/cnc4you/cnc_downloads/cnc4you-magazine/Documents/CNC4you_2-2015_EN.pdf

[18] *Output on an external device: SINUMERIK 828D, 840D sl* [online]. 2014 [cit. 2016-05-20]. Dostupné z: https://cache.industry.siemens.com/dl/files/867/90277867/att_6997/v1/90277867_output_on_an_external_device_doku_v10_en.pdf

[19] *SINUMERIK 840D sl / 828D Synchronized actions: Function Manual* [online]. 2012 [cit. 2016-05-20]. Dostupné z: https://cache.industry.siemens.com/dl/files/968/64932968/att_73756/v1/FBSY_0712_en_en-US.pdf

[20] *Converting string series to float list in python* [online]. In: . 2010 [cit. 2016-05-20]. Dostupné z: http://stackoverflow.com/questions/4004550/converting-string-series-to-float-list-in-python

[21] *Time access and conversions* [online]. [cit. 2016-05-20]. Dostupné z: https://docs.python.org/2/library/time.html

[22] *While loops* [online]. 2012 [cit. 2016-05-20]. Dostupné z: https://wiki.python.org/moin/WhileLoop

[23] *More Control Flow Tools* [online]. [cit. 2016-05-20]. Dostupné z: https://docs.python.org/2/tutorial/controlflow.html

[24] *How to define two-dimensional array in python* [online]. [cit. 2016-05-20]. Dostupné z: http://stackoverflow.com/questions/6667201/how-to-define-two-dimensional-array-in-python

[25] STRACHOTA, Pavel. *Geometrické transformace pomocí matic* [online]. 2010 [cit. 2016-05-20]. Dostupné z: http://saint-paul.fjfi.cvut.cz/base/sites/default/files/POGR/POGR2/07.maticove_transformace.pdf

[26] *Download SinuTrain for SINUMERIK Operate V4.7 - Basic* [online]. [cit. 2016-05-20]. Dostupné z: http://www.industry.siemens.com/topics/global/en/cnc4you/cnc_downloads/sinutrain_downloads/Pages/download-preview-version-sinutrain-for-sinumerik-operate-v4-7-basic.aspx

[27] SMID, Peter. *CNC Programming Handbook*. Avenue of the Americas, New York: Industrial Press Inc.

[28] *Programmlaufzeit ermitteln OHNE $AC_TIMER* [online]. 2009 [cit. 2016-05-20]. Dostupné z: https://support.industry.siemens.com/tf/ww/de/posts/programmlaufzeit-ermitteln-ohne-ac-timer/33998/?page=0&pageSize=10

7 SEZNAM OBRÁZKŠ , TABULEK A ZKRATEK

Zkratka	**Anglický význam**	**Ţeský pŘeklad**
IC.IDO	I see - I do	Já vidím, já dČlám
PLC	Programmable logic controller	Programovatelný logický automat
TLS	Transport layer security	Protokoly pro bezpeţnou komunikaci
SSL	Secure sockets layer	po internetu
SMS	Short message service	Sluţba krátkých textových zpráv
NC	Numerical control	ŢíslicovéŘízení
CAD	Computer aided design	PoţítaţemŘízení podporované projektování
CAM	Computer aided manufacturing	PoţítaţovápodporaŘízení obrábČní
MMC	Man-machine communication	Komunikace mezi ţlovČkem a strojem
NKC	Numerical control kernel	Jádro ţíslicového Řízení

8 Př ÍLOHY

PŘíloha A: IDO skript

PŘíloha B: G-kód na mĚření polohy

PŘíloha C: G-kód pro posuv

PŘíloha D: IC.IDO scéna scene_object.ido (na CD)

PŘíloha E: Model ŘÁdeskaň ve formátu programu CATIA (na CD)

PŚloha A: IDO skript

```
# moving with catia model "deska" (data from one matrix)
import time
import sys
#====================================================================
# coordinate system access
from VDP.IDOKernel.scene.coordinateSystem import
JC_IDOCoordinateSystemList
# common includes
from VDP.IDOKernel.scene.sceneTree import JC_IDOSceneObjectManager
# transformation includes
from VDP.IDOExtern.PlatformDependent.gmtl.inc.gmtl import
JC_Matrix_F_4_4
from de.icido.java.typeMappers import Matrix_F_4_4_Conversion
from jarray import zeros, array
from VDPTypes import MatrixF44;
# math includes
from math import sin,cos,pi
#====================================================================
# opens and converts position data
# g vector of times and positions from the matrix
f=open ('C:\Users\JiriH\Documents\ICI.DO_scripts\Movement_test\
X_position.txt' , 'r')
g=[map(float,line.split(' ')) for line in f]
print g

g_dimensions=len(g)
t_overall=g[g_dimensions-1][0]

soList = JC_IDOSceneObjectManager.getSingleton().
getSceneObjectListByName('Deska')
sceneObject = soList.getFirstSceneObject()

attributeId = JC_IDOCoordinateSystemList.getAttributeAttributeId()
coordinateSystemList = sceneObject.getAttribute(attributeId)

def attach():
     row=0
     userDefinedCurrentCoordinateD =
coordinateSystemList.getCurrent().getCoordinateSystem()
     dpData = userDefinedCurrentCoordinateD.getData();
```

```
print dpData
fData = []
for i in range(0, 4):
      fData.append([])
      for j in range(0, 4):
            fData[i].append( float( dpData.
            getDouble( i + j * 4 ) ) )

#print fData
userDefinedCurrentCoordinate = MatrixF44( fData );
userDefinedCurrentCoordinateInv =
userDefinedCurrentCoordinate.inverse()

cas_zacatku=time.clock()
cas_behu=0
row=0
while cas_behu < t_overall:
      cas_behu=time.clock()-cas_zacatku
      if cas_behu >= g[row][0]:
            if cas_behu == 0:
                  a=(g[row][1])/1000
            else:
                  a=(g[row][1]-g[row-1][1])/1000
            # create translation matrix
            matrixTransJC = JC_Matrix_F_4_4()
            matrixTransJC.set(
                        1, 0, 0, a,
                        0, 1, 0, 0,
                        0, 0, 1, 0,
                        0, 0, 0, 1 )
            matrixTrans = Matrix_F_4_4_Conversion.instance.
            toExternal(matrixTransJC)

            print g[row][0]


            matrixWS = sceneObject.getTransformWS()
            matrixTemp1 = userDefinedCurrentCoordinateInv.
            times(matrixWS)
            matrixTemp2 = matrixTrans.times(matrixTemp1)
```

```
                matrixTemp3 = userDefinedCurrentCoordinate.
                times(matrixTemp2)

                sceneObject.setTransformBSbyWS(matrixTemp3)
                row=row+1

def detach():
     print "Script detaching ..."

def release():
     print "Script releasing ..."
```

Příloha B: G-kód na měření polohy

```
N1 DEF INT cas_behu=10
N2 DEF INT row
N3 DEF REAL matrix[70,2]
N4 DEF INT radek
N5 DEF REAL start_seconds
N6 DEF INT XPOSITION
N10 DEF REAL seconds
N15 DEF INT TIMER
N20 DEF REAL cas_ted
N25 DEF REAL start_cyklus
N30 start_seconds = $A_SECOND+$A_MINUTE*60+$A_HOUR*3600
N35 start_cyklus=$A_SECOND+$A_MINUTE*60+$A_HOUR*3600
N40 seconds=$A_SECOND+$A_MINUTE*60+$A_HOUR*3600-start_seconds
N45 cas_ted=$A_SECOND+$A_MINUTE*60+$A_HOUR*3600-start_cyklus
N50 CYKLUS:
N70 IF cas_ted>=0.5
N71 seconds=$A_SECOND+$A_MINUTE*60+$A_HOUR*3600-start_seconds
N75 matrix[row,0]=seconds
N80 matrix[row,1]=$AA_IM[X]
N85 row=row+1
N86 start_cyklus=$A_SECOND+$A_MINUTE*60+$A_HOUR*3600
N87 ENDIF
N92 cas_ted=$A_SECOND+$A_MINUTE*60+$A_HOUR*3600-start_cyklus
N95 IF seconds<cas_behu GOTOB CYKLUS

N100 EXTOPEN(XPOSITION,"X_POSITION.TXT","SYN","SHARED")
N110 FOR radek=0 to row-1
N120 WRITE(XPOSITION,"X_POSITION.TXT",<<matrix[radek,0]<<"
"<<matrix[radek,1])
N125 ENDFOR
N200 M30
```

PŚloha C: G-kód pro posuv

```
N1 G54
N5 F100
N10 G1 X500
N15 G1 X0
N20 M30
```