The RISC-V Debug Specification

Editors: Paul Donahue <pdonahue@ventanamicro.com>, Ventana Micro Systems, Tim Newsome <tim@sifive.com>, SiFive, Inc.

Version 20231107, Revised 20231107

Preface

Contributors to all versions of the spec in alphabetical order (please contact editors to suggest corrections): Bruce Ableidinger, Krste Asanović, Peter Ashenden, Allen Baum, Mark Beal, Alex Bradbury, Chuanhua Chang, Yen Hao Chen, Zhong-Ho Chen, Monte Dalrymple, Paul Donahue, Vyacheslav Dyachenko, Ernie Edgar, Peter Egold, Marc Gauthier, Markus Goehrle, Robert Golla, John Hauser, Richard Herveille, Yung-ching Hsiao, Po-wei Huang, Scott Johnson, L. J. Madar, Grigorios Magklis, Daniel Mangum, Alexis Marquet, Jan Matyas, Kai Meinhard, Jean-Luc Nagel, Aram Nahidipour, Rishiyur Nikhil, Gajinder Panesar, Deepak Panwar, Antony Pavlov, Klaus Kruse Pedersen, Ken Pettit, Darius Rad, Joe Rahmeh, Josh Scheid, Vedvyas Shanbhogue, Gavin Stark, Ben Staveley, Wesley Terpstra, Tommy Thorn, Megan Wachs, Jan-Willem van de Waerdt, Philipp Wagner, Stefan Wallentowitz, Ray Van De Walker, Andrew Waterman, Thomas Wicki, Andy Wright, Bryan Wyatt, and Florian Zaruba.

Chapter 1. Introduction

When a design progresses from simulation to hardware implementation, a user's control and understanding of the system's current state drops dramatically. To help bring up and debug low level software and hardware, it is critical to have good debugging support built into the hardware. When a robust OS is running on a core, software can handle many debugging tasks. However, in many scenarios, hardware support is essential.

This document outlines a standard architecture for debug support on RISC-V hardware platforms. This architecture allows a variety of implementations and tradeoffs, which is complementary to the wide range of RISC-V implementations. At the same time, this specification defines common interfaces to allow debugging tools and components to target a variety of hardware platforms based on the RISC-V ISA.

System designers may choose to add additional hardware debug support, but this specification defines a standard interface for common functionality.

1.1. Terminology

advanced feature
An advanced feature for advanced users. Most users will not be able to take advantage of it.

AMO Atomic Memory Operation.

BYPASS JTAG instruction that selects a single bit data register, also called BYPASS.

component A RISC-V core, or other part of a hardware platform. Typically all components will be connected to a single system bus.

CSR Control and Status Register.

DM Debug Module (see Chapter Chapter 3).

DMI Debug Module Interface (see Section Section 3.1).

DR JTAG Data Register.

DTM Debug Transport Module (see Section Chapter 15).

DXLEN Debug XLEN, which is the widest XLEN a hart supports, ignoring the current value of in .

essential feature An essential feature must be present in order for debug to work correctly.

GPR General Purpose Register.

hardware platform A single system consisting of one or more *components*.

hart A hardware thread in a RISC-V core.

IDCODE 32-bit Identification CODE, and a JTAG instruction that returns the IDCODE value.

IR JTAG Instruction Register.

JTAG Refers to work done by IEEE's Joint Test Action Group, described in IEEE 1149.1.

legacy feature A legacy feature should only be implemented to support legacy hardware that is present in a system.

Minimal RISC-V Debug Specification A subset of the full Debug Specification that allows for very small implementations. See Chapter Chapter 3.

NAPOT Naturally Aligned Power-Of-Two.

NMI Non-Maskable Interrupt.

physical address address that is directly usable on the system bus.

recommended feature A recommended feature is not required for debug to work correctly, but it is so useful that it should not be omitted without good reason.

SBA System Bus Access (see Section Section 3.10).

specialized feature A specialized feature, that only makes sense in the context of some specific hardware.

TAP Test Access Port, defined in IEEE 1149.1.

TM Trigger Module (see Section [trigger]).

virtual address An address as a hart sees it. If the hart is using address translation this may be different from the physical address. If there is no translation then it will be the same.

xepc The exception program counter CSR (e.g.) that is appropriate for the mode being trapped to.

1.2. Context

This specification attempts to support all RISC-V ISA extensions that have, roughly, been ratified through the first half of 2023. In particular, though, this specification specifically addresses features in the following extensions:

1. A
2. C
3. D
4. F
5. H
6. Sm1p13
7. Ss1p13
8. Smstateen
9. V
10. Zawrs
11. Zcmp
12. Zicbom

13. Zicboz
14. Zicbop

1.2.1. Versions

Version 0.13 of this document was ratified by the RISC-V Foundation's board. Versions 0.13.*x* are bug fix releases to that ratified specification.

Version 0.14 was a working version that was never officially ratified.

Version 1.0 is almost entirely forwards and backwards compatible with Version 0.13.

1.2.1.1. Bugfixes from 0.13 to 1.0

Changes that fix a bug in the spec:

1. Fix order of operations described in [sbdata0]. #392
2. Resume ack is set after resume, in Section 3.5. #400
3. [sselect] applies to [svalue] . #402
4. [mte] only applies when action=0. #411
5. [aamsize] does not affect Argument Width. #420
6. Clarify that harts halt out of reset if [haltreq] =1. #419

1.2.1.2. Incompatible Changes from 0.13 to 1.0

Changes that are not backwards-compatible. Debuggers or hardware implementations that implement 0.13 will have to change something in order to implement 1.0:

1. Make haltsum0 optional if there is only one hart. #505
2. System bus autoincrement only happens if an access actually takes place. ([sbdata0]) #507
3. Bump [version] to 3. #512 , Require debugger to poll [dmactive] after lowering it. #566
4. Add [pending] to [icount] . #574
5. When a selected trigger is disabled, [tdata2] and [tdata3] can be written with any value supported by any of the types this trigger supports. #721
6. [tcontrol] fields only apply to breakpoint traps, not any trap. #723
7. If [version] is greater than 0, then [hit0] (previously called [mcontrol].`hit`) now contains 0 when a trigger fires more than one instruction after the instruction that matched. (This information is now reflected in .) #795
8. If [version] is greater than 0, then bit 20 of [mcontrol16] is no longer used for timing information. (Previously the bit was called [mcontrol].`timing`.) #807
9. If [version] is greater than 0, then the encodings of [size] for sizes greater than 64 bit have changed. #807

1.2.1.3. Minor Changes from 0.13 to 1.0

Changes that slightly modify defined behavior. Technically backwards incompatible, but unlikely to be noticeable:

1. [stopcount] only applies to hart-local counters. #405
2. [version] may be invalid when [dmactive]=0. #414
3. Address triggers ([mcontrol]) may fire on any accessed address. #421
4. All trigger registers ([csrTrigger]) are optional. #431
5. When extending IR, [bypass] still is all ones. #437
6. [ebreaks] and [ebreaku] are WARL. #458
7. NMIs are disabled by [stepie]. #465
8. R/W1C fields should be cleared by writing every bit high. #472
9. Specify trigger priorities in [priority] relative to exceptions. #478
10. Time may pass before [dmactive] becomes high. #500
11. Clear MPRV when resuming into lower privilege mode. #503
12. Halt state may not be preserved across reset. #504
13. Hardware should clear trigger action when [dmode] is cleared and action is 1. #501
14. Change quick access exceptions to halt the target in [acQuickaccess]. #585
15. Writing 0 to [tdata1] forces a state where [tdata2] and [tdata3] are writable. #598
16. Solutions to deal with reentrancy in [nativetrigger] prevent triggers from *matching*, not merely *firing*. This primarily affects behavior. #722
17. Attempts to access an unimplemented CSR raise an illegal instruction exception. #791

1.2.1.4. New Features from 0.13 to 1.0

New backwards-compatible feature that did not exist before:

1. Add halt groups and external triggers in Section 3.6. #404
2. Reserve some DMI space for non-standard use. See [custom], and [custom0] through . #406
3. Reserve trigger [type] values for non-standard use. #417
4. Add [nmi] bit to [itrigger]. #408 and #709
5. Recommend matching on every accessed address. #449
6. Add resume groups in Section 3.6. #506
7. Add [relaxedpriv] . #536
8. Move [scontext], renaming original to [mscontext], and create [hcontext]. #535
9. Add [mcontrol6], deprecating [mcontrol]. #538
10. Add hypervisor support: [ebreakvs], [ebreakvu], [v], [hcontext], [mcontrol], [mcontrol6], and [priv]. #549
11. Optionally make [anyunavail] and [allunavail] sticky, controlled by [stickyunavail]. #520
12. Add [tmexttrigger] to support trigger module external trigger inputs. #543
13. Describe [mcontrol] and [mcontrol6] behavior with atomic instructions. #561
14. Trigger hit bits must be set on fire, may be set on match. #593
15. Add [sbytemask] and [sbytemask] to [textra32] and [textra64]. #588

16. Allow debugger to request harts stay alive with keepalive bit in [keepalive]. #592
17. Add [ndmresetpending] to allow a debugger to determine when ndmreset is complete. #594
18. Add [intctl] to support triggers from an interrupt controller. #599

1.2.1.5. Incompatible Changes During 1.0 Stable

Backwards-incompatible changes between two versions that are both called 1.0 stable.

1. [nmi] was moved from [etrigger] to [itrigger], and is now subject to the mode bits in that trigger.
2. #728 introduced Message Registers, which were later removed in #878.
3. It may not be possible to read the contents of the Program Buffer using the `progbuf` registers. #731
4. [tcontrol] fields apply to all traps, not just breakpoint traps. This reverts #723. #880

1.3. About This Document

1.3.1. Structure

This document contains two parts. The main part of the document is the specification, which is given in the numbered chapters. The second part of the document is a set of appendices. The information in the appendices is intended to clarify and provide examples, but is not part of the actual specification.

1.3.2. ISA vs. non-ISA

This specification contains both ISA and non-ISA parts. The ISA parts define self-contained ISA extensions. The other parts of the document describe the non-ISA external debug extension. Chapters whose contents are solely one or the other are labeled as such in their title. Chapters without such a label apply to both ISA and non-ISA.

1.3.3. Register Definition Format

All register definitions in this document follow the format shown below. A simple graphic shows which fields are in the register. The upper and lower bit indices are shown to the top left and top right of each field. The total number of bits in the field are shown below it.

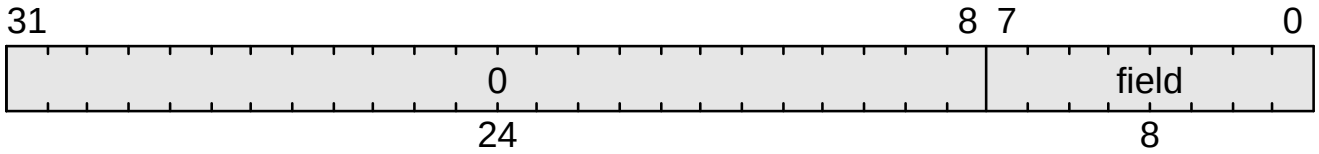
After the graphic follows a table which for each field lists its name, description, allowed accesses, and reset value. The allowed accesses are listed in [access]. The reset value is either a constant or "Preset." The latter means it is an implementation-specific legal value.

Parts of the register which are currently unused are labeled with the number 0. Software must only write 0 to those fields, and ignore their value while reading. Hardware must return 0 when those fields are read, and ignore the value written to them.

This behavior enables us to use those fields later without having to increase the values in the version fields.

Names of registers and their fields are hyperlinks to their definition, and are also listed in the index on page [singlestep].

1.3.3.1. Long Name (shortname, `at 0x123`)



Field	Description	Access	Reset
field	Description of what this field is used for.	R/W	15

R	Read-only.
R/W	Read/Write.
R/W1C	Read/Write ones to Clear. Writing 0 to every bit as no effect. Writing 1 to every bit clears the field. The result of other writes is undefined.
WARZ	Write any, read zero. A debugger may write any value. When read this field returns 0.
W1	Write-only. Only writing 1 has an effect. When read the returned value should be 0.
WARL	Write any, read legal. A debugger may write any value. If a value is unsupported, the implementation converts the value to one that is supported.

Table 1. Register Access Abbreviations

1.4. Background

There are several use cases for dedicated debugging hardware, both in native debug and external debug. Native debug (sometimes called self-hosted debug) refers to debug software running on a RISC-V platform which debugs the same platform. The optional Trigger Module provides features that are useful for native debug. External debug refers to debug software running somewhere else, debugging the RISC-V platform via a debug transport like JTAG. The entire document provides features that are useful for external debug.

This specification addresses the use cases listed below. Implementations can choose not to implement every feature, which means some use cases might not be supported.

- Accessing hardware on a hardware platform without a working CPU. (External debug.)
- Bootstrapping a hardware platform to test, configure, and program components before there is any executable code path in the hardware platform. (External debug.)
- Debugging low-level software in the absence of an OS or other software. (External debug.)
- Debugging issues in the OS itself. (External or native debug.)
- Debugging processes running on an OS. (Native or external debug.)

1.5. Supported Features

The debug interface described in this specification supports the following features:

1. All hart registers (including CSRs) can be read/written.
2. Memory can be accessed either from the hart's point of view, through the system bus directly, or both.

3. RV32, RV64, and future RV128 are all supported.
4. Any hart in the hardware platform can be independently debugged.
5. A debugger can discover almost [1] everything it needs to know itself, without user configuration.
6. Each hart can be debugged from the very first instruction executed.
7. A RISC-V hart can be halted when a software breakpoint instruction is executed.
8. Hardware single-step can execute one instruction at a time.
9. Debug functionality is independent of the debug transport used.
10. The debugger does not need to know anything about the microarchitecture of the harts it is debugging.
11. Arbitrary subsets of harts can be halted and resumed simultaneously. (Optional)
12. Arbitrary instructions can be executed on a halted hart. That means no new debug functionality is needed when a core has additional or custom instructions or state, as long as there exist programs that can move that state into GPRs. (Optional)
13. Registers can be accessed without halting. (Optional)
14. A running hart can be directed to execute a short sequence of instructions, with little overhead. (Optional)
15. A system bus manager allows memory access without involving any hart. (Optional)
16. A RISC-V hart can be halted when a trigger matches the PC, read/write address/data, or an instruction opcode. (Optional)
17. Harts can be grouped, and harts in the same group will all halt when any of them halts. These groups can also react to or notify external triggers. (Optional)

This document does not suggest a strategy or implementation for hardware test, debugging or error detection techniques. Scan, built-in self test (BIST), etc. are out of scope of this specification, but this specification does not intend to limit their use in RISC-V systems.

It is possible to debug code that uses software threads, but there is no special debug support for it.

[1] Notable exceptions include information about the memory map and peripherals.

Chapter 2. System Overview

Chapter 2 shows the main components of Debug Support. Blocks shown in dotted lines are optional.

RISC-V Debug System Overview


linejoin="miter" fill="none" stroke="#000000" d="M5174.9 858.297H7299.5V3391.4268H5174.9Z"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M5023.14 1000.05H7147.7307V3543.1602H5023.14Z" fill="#ffffff" fill-rule="evenodd"/> <path transform="matrix(.1,0,0,-.1,0,796)" stroke-width="2.8346" stroke-linecap="butt" stroke-miterlimit="10" stroke-linejoin="miter" fill="none" stroke="#ffffff" d="M5023.14 1000.05H7147.7307V3543.1602H5023.14Z"/> <path transform="matrix(.1,0,0,-.1,0,796)" stroke-width="30.3515" stroke-linecap="butt" stroke-dasharray="60.7027" stroke-miterlimit="10" stroke-linejoin="miter" fill="none" stroke="#000000" d="M5023.14 1000.05H7147.7307V3543.1602H5023.14Z"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M4871.4 1141.82H6995.99V3694.9H4871.4Z" fill="#ffffff" fill-rule="evenodd"/> <path transform="matrix(.1,0,0,-.1,0,796)" stroke-width="2.8346" stroke-linecap="butt" stroke-miterlimit="10" stroke-linejoin="miter" fill="none" stroke="#ffffff" d="M4871.4 1141.82H6995.99V3694.9H4871.4Z"/> <path transform="matrix(.1,0,0,-.1,0,796)" stroke-width="30.3515" stroke-linecap="butt" stroke-miterlimit="10" stroke-linejoin="miter" fill="none" stroke="#000000" d="M4871.4 1141.82H6995.99V3694.9H4871.4Z"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M4949.86 4051.04C4953.65 4049.79 4957.34 4047.12 4960.94 4043.01 4964.55 4038.9 4968.17 4033.27 4971.79 4026.11L4988.73 3991.2H4970.73L4953.9 4023.94C4949.67 4032.5 4945.57 4038.17 4941.61 4040.96 4937.66 4043.77 4932.25 4045.17 4925.41 4045.17H4906.6V3991.2H4887.82V4120.27H4928.3C4943.09 4120.27 4954.12 4117.15 4961.38 4110.91 4968.66 4104.68 4972.3 4095.29 4972.3 4082.72 4972.3 4074.53 4970.39 4067.72 4966.55 4062.3 4962.73 4056.89 4957.17 4053.14 4949.86 4051.04M4906.6 4106.19V4059.25H4928.3C4936.64 4059.25 4942.92 4061.25 4947.15 4065.23 4951.41 4069.21 4953.53 4075.07 4953.53 4082.79 4953.53 4090.54 4951.41 4096.37 4947.15 4100.28 4942.92 4104.22 4936.64 4106.19 4928.3 4106.19H4906.6"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M5013.85 4120.27H5032.62V3991.2H5013.85V4120.27"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M5147.44 4115.57V4099.14C5140.79 4102.3 5134.52 4104.64 5128.63 4106.19 5122.74 4107.75 5117.04 4108.53 5111.54 4108.53 5102.01 4108.53 5094.65 4106.72 5089.47 4103.11 5084.31 4099.49 5081.73 4094.34 5081.73 4087.67 5081.73 4082.07 5083.41 4077.84 5086.76 4074.98 5090.13 4072.15 5096.51 4069.85 5105.89 4068.09L5116.23 4065.96C5129.31 4063.47 5138.96 4059.06 5145.16 4052.73 5151.38 4046.42 5154.48 4037.96 5154.48 4027.35 5154.48 4014.71 5150.23 4005.13 5141.72 3998.61 5133.24 3992.11 5120.8 3988.85 5104.39 3988.85 5098.21 3988.85 5091.62 3989.64 5084.63 3991.2 5077.66 3992.79 5070.44 3995.14 5062.96 3998.24V4017.01C5070.12 4012.34 5077.13 4008.82 5083.97 4006.45 5090.84 4004.11 5097.59 4002.93 5104.21 4002.93 5114.28 4002.93 5122.04 4004.95 5127.49 4008.98 5132.97 4013.02 5135.71 4018.76 5135.71 4026.21 5135.71 4032.72 5133.8 4037.8 5129.98 4041.47 5126.17 4045.16 5119.91 4047.92 5111.21 4049.76L5100.8 4051.77C5087.48 4054.32 5077.84 4058.3 5071.87 4063.73 5065.93 4069.18 5062.96 4076.74 5062.96 4086.42 5062.96 4097.64 5067.04 4106.48 5075.21 4112.93 5083.39 4119.39 5094.66 4122.61 5109.01 4122.61 5115.17 4122.61 5121.44 4122.03 5127.82 4120.85 5134.23 4119.68 5140.77 4117.92 5147.44 4115.57"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M5281.61 4110.88V4092.11C5275.65 4097.61 5269.29 4101.71 5262.55 4104.43 5255.8 4107.16 5248.64 4108.53 5241.06 4108.53 5226.1 4108.53 5214.65 4104 5206.71 4094.93 5198.76 4085.88 5194.79 4072.81 5194.79 4055.7 5194.79 4038.63 5198.76 4025.57 5206.71 4016.5 5214.65 4007.46 5226.1 4002.93 5241.06 4002.93 5248.64 4002.93 5255.8 4004.29 5262.55 4007 5269.29 4009.74 5275.65 4013.86 5281.61 4019.36V4000.59C5275.41 3996.68 5268.84 3993.74 5261.93 3991.79 5255.01 3989.83 5247.69 3988.85 5239.96 3988.85 5220.14 3988.85 5204.52 3994.82 5193.1 4006.75 5181.71 4018.7 5176.02 4035.02 5176.02 4055.7 5176.02 4076.43 5181.71 4092.75 5193.1 4104.68 5204.52 4116.64 5220.14 4122.61 5239.96 4122.61 5247.79 4122.61 5255.15 4121.63 5262.07 4119.68 5269.02 4117.72 5275.53 4114.79 5281.61 4110.88"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M5302.75 4047.52H5349.68V4033.44H5302.75V4047.52"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M5400.09 3991.2 5350.22 4120.27H5368.66L5410.06 4011.59 5451.53 4120.27H5469.9L5420.11 3991.2H5400.09"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M5644.1 4110.88V4092.11C5638.14 4097.61 5631.78 4101.71 5625.04 4104.43 5618.29 4107.16 5611.13 4108.53 5603.55 4108.53 5588.59

4108.53 5577.14 4104 5569.19 4094.93 5561.25 4085.88 5557.28 4072.81 5557.28 4055.7 5557.28 4038.63 5561.25 4025.57 5569.19 4016.5 5577.14 4007.46 5588.59 4002.93 5603.55 4002.93 5611.13 4002.93 5618.29 4004.29 5625.04 4007 5631.78 4009.74 5638.14 4013.86 5644.1 4019.36V4000.59C5637.89 3996.68 5631.33 3993.74 5624.41 3991.79 5617.5 3989.83 5610.18 3988.85 5602.45 3988.85 5582.63 3988.85 5567 3994.82 5555.59 4006.75 5544.2 4018.7 5538.5 4035.02 5538.5 4055.7 5538.5 4076.43 5544.2 4092.75 5555.59 4104.68 5567 4116.64 5582.63 4122.61 5602.45 4122.61 5610.27 4122.61 5617.64 4121.63 5624.56 4119.68 5631.5 4117.72 5638.02 4114.79 5644.1 4110.88"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M5709.86 4078.03C5701.15 4078.03 5694.27 4074.68 5689.21 4067.98 5684.18 4061.28 5681.66 4052.11 5681.66 4040.48 5681.66 4028.84 5684.16 4019.68 5689.18 4012.98 5694.21 4006.28 5701.11 4002.93 5709.86 4002.93 5718.51 4002.93 5725.35 4006.29 5730.39 4013.02 5735.45 4019.74 5737.98 4028.89 5737.98 4040.48 5737.98 4052.02 5735.45 4061.16 5730.39 4067.91 5725.35 4074.65 5718.51 4078.03 5709.86 4078.03M5709.82 4092.11C5723.7 4092.11 5734.61 4087.54 5742.52 4078.39 5750.45 4069.27 5754.41 4056.64 5754.41 4040.48 5754.41 4024.37 5750.45 4011.73 5742.52 4002.57 5734.61 3993.43 5723.7 3988.85 5709.82 3988.85 5695.89 3988.85 5684.97 3993.43 5677.07 4002.57 5669.18 4011.73 5665.23 4024.37 5665.23 4040.48 5665.23 4056.64 5669.18 4069.27 5677.07 4078.39 5684.97 4087.54 5695.89 4092.11 5709.82 4092.11"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M5839.75 4075.68C5837.98 4076.48 5836.08 4077.07 5834.02 4077.44 5831.97 4077.83 5829.7 4078.03 5827.21 4078.03 5818.41 4078.03 5811.65 4075.08 5806.93 4069.19 5802.21 4063.32 5799.85 4054.89 5799.85 4043.89V3991.2H5783.43V4089.76H5799.85V4075.68C5803.13 4081.23 5807.39 4085.35 5812.65 4088.04 5817.9 4090.75 5824.29 4092.11 5831.79 4092.11 5832.86 4092.11 5834.05 4092.11 5835.34 4092.11 5836.64 4092.11 5838.08 4092.11 5839.67 4092.11L5839.75 4075.68"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M5937.99 4045.46V4038.13H5862.9C5863.61 4026.67 5866.96 4017.93 5872.95 4011.92 5878.94 4005.93 5887.27 4002.93 5897.95 4002.93 5904.16 4002.93 5910.18 4003.71 5915.99 4005.28 5921.81 4006.84 5927.58 4009.19 5933.3 4012.32V3998.24C5927.53 3995.18 5921.62 3992.85 5915.55 3991.24 5909.52 3989.65 5903.39 3988.85 5897.18 3988.85 5881.59 3988.85 5869.23 3993.41 5860.11 4002.53 5851.02 4011.67 5846.47 4024.03 5846.47 4039.6 5846.47 4055.71 5850.84 4068.48 5859.56 4077.91 5868.29 4087.38 5880.06 4092.11 5894.88 4092.11 5908.17 4092.11 5918.68 4087.93 5926.41 4079.57 5934.13 4071.21 5937.99 4059.84 5937.99 4045.46M5921.57 4049.87C5921.45 4058.42 5918.94 4065.25 5914.05 4070.36 5909.19 4075.47 5902.75 4078.03 5894.73 4078.03 5885.63 4078.03 5878.35 4075.56 5872.88 4070.62 5867.42 4065.68 5864.28 4058.73 5863.45 4049.76L5921.57 4049.87"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M318.688 2170.12H2746.798V5508.77H318.688Z" fill="#ffffff" fill-rule="evenodd"/> <path transform="matrix(.1,0,0,-.1,0,796)" stroke-width="2.8346" stroke-linecap="butt" stroke-miterlimit="10" stroke-linejoin="miter" fill="none" stroke="#ffffff" d="M318.688 2170.12H2746.798V5508.77H318.688Z"/> <path transform="matrix(.1,0,0,-.1,0,796)" stroke-width="30.3515" stroke-linecap="butt" stroke-miterlimit="10" stroke-linejoin="miter" fill="none" stroke="#000000" d="M318.688 2170.12H2746.798V5508.77H318.688Z"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M657.402 5320.24V5219.34H678.523C696.367 5219.34 709.43 5223.39 717.719 5231.51 726.004 5239.63 730.148 5252.43 730.148 5269.94 730.148 5287.32 726.004 5300.04 717.719 5308.11 709.43 5316.2 696.367 5320.24 678.523 5320.24H657.402M638.629 5334.32H675.844C701.023 5334.32 719.492 5329.12 731.246 5318.7 743.031 5308.31 748.922 5292.06 748.922 5269.94 748.922 5247.67 743.004 5231.31 731.176 5220.88 719.344 5210.46 700.902 5205.26 675.844 5205.26H638.629V5334.32"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M862.578 5259.52V5252.19H787.488C788.195 5240.73 791.547 5231.99 797.535 5225.97 803.523 5219.98 811.859 5216.99 822.539 5216.99 828.75 5216.99 834.762 5217.77 840.582 5219.34 846.398 5220.9 852.168 5223.25 857.887 5226.38V5212.3C852.117 5209.24 846.203 5206.91 840.141 5205.29 834.102 5203.7 827.98 5202.91 821.77 5202.91 806.176 5202.91 793.82 5207.47 784.699 5216.59 775.609 5225.73 771.063 5238.09 771.063 5253.66 771.063 5269.77 775.426 5282.54 784.152 5291.97 792.879 5301.43 804.648 5306.16 819.461 5306.16 832.758

5306.16 843.27 5301.98 850.992 5293.62 858.719 5285.26 862.578 5273.89 862.578 5259.52M846.152 5263.92C846.031 5272.48 843.527 5279.31 838.637 5284.42 833.773 5289.53 827.332 5292.08 819.313 5292.08 810.219 5292.08 802.938 5289.61 797.461 5284.68 792.008 5279.74 788.867 5272.79 788.039 5263.81L846.152 5263.92"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M960.359 5254.54C960.359 5266.25 957.953 5275.42 953.137 5282.07 948.348 5288.75 941.758 5292.08 933.375 5292.08 924.988 5292.08 918.391 5288.75 913.574 5282.07 908.785 5275.42 906.387 5266.25 906.387 5254.54 906.387 5242.83 908.785 5233.64 913.574 5226.96 918.391 5220.31 924.988 5216.99 933.375 5216.99 941.758 5216.99 948.348 5220.31 953.137 5226.96 957.953 5233.64 960.359 5242.83 960.359 5254.54M906.387 5289.73C909.711 5295.29 913.918 5299.4 919 5302.09 924.086 5304.8 930.16 5306.16 937.223 5306.16 948.957 5306.16 958.48 5301.42 965.789 5291.94 973.121 5282.45 976.789 5269.98 976.789 5254.54 976.789 5239.09 973.121 5226.62 965.789 5217.14 958.48 5207.65 948.957 5202.91 937.223 5202.91 930.16 5202.91 924.086 5204.25 919 5206.94 913.918 5209.66 909.711 5213.79 906.387 5219.34V5205.26H889.961V5341.36H906.387V5289.73"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1005.36 5243.39V5303.82H1021.79V5244.01C1021.79 5235.02 1023.57 5228.26 1027.14 5223.74 1030.71 5219.24 1036.08 5216.99 1043.24 5216.99 1051.82 5216.99 1058.6 5219.69 1063.59 5225.09 1068.58 5230.52 1071.07 5237.9 1071.07 5247.24V5303.82H1087.5V5205.26H1071.07V5219.34C1067.18 5213.79 1062.66 5209.66 1057.5 5206.94 1052.37 5204.25 1046.41 5202.91 1039.61 5202.91 1028.39 5202.91 1019.87 5206.34 1014.05 5213.21 1008.26 5220.11 1005.36 5230.16 1005.36 5243.39M1045.92 5306.16V5306.16"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1183.41 5255.67C1183.41 5267.23 1181.04 5276.2 1176.3 5282.55 1171.58 5288.91 1164.95 5292.08 1156.39 5292.08 1147.89 5292.08 1141.26 5288.91 1136.52 5282.55 1131.8 5276.2 1129.44 5267.23 1129.44 5255.67 1129.44 5244.16 1131.8 5235.23 1136.52 5228.87 1141.26 5222.52 1147.89 5219.34 1156.39 5219.34 1164.95 5219.34 1171.58 5222.52 1176.3 5228.87 1181.04 5235.23 1183.41 5244.16 1183.41 5255.67M1199.84 5217.98C1199.84 5201.02 1196.16 5188.4 1188.8 5180.14 1181.47 5171.86 1170.24 5167.71 1155.11 5167.71 1149.48 5167.71 1144.2 5168.1 1139.23 5168.88 1134.27 5169.64 1129.44 5170.82 1124.75 5172.4V5188.83C1129.39 5186.43 1133.98 5184.66 1138.5 5183.51 1143.05 5182.36 1147.66 5181.79 1152.36 5181.79 1162.75 5181.79 1170.52 5184.54 1175.68 5190.04 1180.84 5195.54 1183.41 5203.84 1183.41 5214.94V5221.68C1180.14 5216.18 1175.95 5212.06 1170.84 5209.33 1165.73 5206.61 1159.61 5205.26 1152.47 5205.26 1140.64 5205.26 1131.11 5209.85 1123.87 5219.04 1116.63 5228.26 1113.02 5240.47 1113.02 5255.67 1113.02 5270.93 1116.63 5283.15 1123.87 5292.34 1131.11 5301.55 1140.64 5306.16 1152.47 5306.16 1159.61 5306.16 1165.73 5304.79 1170.84 5302.05 1175.95 5299.34 1180.14 5295.23 1183.41 5289.73V5303.82H1199.84V5217.98"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1292.8 5334.32H1320.45L1352.61 5247.35 1384.95 5334.32H1412.48V5205.26H1393.71V5318.74L1361.22 5231.07H1344.06L1311.58 5318.74V5205.26H1292.8V5334.32"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1486.85 5292.08C1478.15 5292.08 1471.27 5288.73 1466.21 5282.04 1461.17 5275.34 1458.65 5266.17 1458.65 5254.54 1458.65 5242.9 1461.16 5233.73 1466.17 5227.04 1471.21 5220.34 1478.1 5216.99 1486.85 5216.99 1495.5 5216.99 1502.35 5220.35 1507.38 5227.07 1512.45 5233.8 1514.97 5242.95 1514.97 5254.54 1514.97 5266.07 1512.45 5275.21 1507.38 5281.96 1502.35 5288.71 1495.5 5292.08 1486.85 5292.08M1486.81 5306.16C1500.7 5306.16 1511.6 5301.59 1519.52 5292.45 1527.44 5283.33 1531.4 5270.7 1531.4 5254.54 1531.4 5238.43 1527.44 5225.79 1519.52 5216.63 1511.6 5207.48 1500.7 5202.91 1486.81 5202.91 1472.88 5202.91 1461.96 5207.48 1454.07 5216.63 1446.18 5225.79 1442.23 5238.43 1442.23 5254.54 1442.23 5270.7 1446.18 5283.33 1454.07 5292.45 1461.96 5301.59 1472.88 5306.16 1486.81 5306.16"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1623.78 5289.73V5341.36H1640.21V5205.26H1623.78V5219.34C1620.46 5213.79 1616.25 5209.66 1611.17 5206.94 1606.08 5204.25 1599.97 5202.91 1592.83 5202.91 1581.17 5202.91 1571.68 5207.65 1564.34 5217.14 1557.04 5226.62 1553.38 5239.09 1553.38 5254.54 1553.38 5269.98 1557.04 5282.45 1564.34 5291.94 1571.68 5301.42 1581.17 5306.16 1592.83 5306.16 1599.97 5306.16 1606.08 5304.8 1611.17 5302.09 1616.25 5299.4 1620.46 5295.29 1623.78 5289.73M1569.81 5254.54C1569.81 5242.83 1572.2 5233.64 1576.99 5226.96 1581.79 5220.31 1588.37 5216.99 1596.76 5216.99 1605.14 5216.99 1611.74


5220.31 1616.56 5226.96 1621.37 5233.64 1623.78 5242.83 1623.78 5254.54 1623.78 5266.25 1621.37 5275.42 1616.56 5282.07 1611.74 5288.75 1605.14 5292.08 1596.76 5292.08 1588.37 5292.08 1581.79 5288.75 1576.99 5282.07 1572.2 5275.42 1569.81 5266.25 1569.81 5254.54"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1675.82 5243.39V5303.82H1692.25V5244.01C1692.25 5235.02 1694.04 5228.26 1697.6 5223.74 1701.17 5219.24 1706.54 5216.99 1713.7 5216.99 1722.28 5216.99 1729.06 5219.69 1734.05 5225.09 1739.04 5230.52 1741.53 5237.9 1741.53 5247.24V5303.82H1757.95V5205.26H1741.53V5219.34C1737.64 5213.79 1733.12 5209.66 1727.96 5206.94 1722.83 5204.25 1716.86 5202.91 1710.07 5202.91 1698.85 5202.91 1690.33 5206.34 1684.51 5213.21 1678.72 5220.11 1675.82 5230.16 1675.82 5243.39M1716.38 5306.16V5306.16"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1790.52 5341.36H1806.94V5205.26H1790.52V5341.36"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1925.26 5259.52V5252.19H1850.17C1850.88 5240.73 1854.22 5231.99 1860.21 5225.97 1866.2 5219.98 1874.54 5216.99 1885.22 5216.99 1891.43 5216.99 1897.44 5217.77 1903.26 5219.34 1909.08 5220.9 1914.85 5223.25 1920.57 5226.38V5212.3C1914.8 5209.24 1908.88 5206.91 1902.82 5205.29 1896.78 5203.7 1890.66 5202.91 1884.45 5202.91 1868.86 5202.91 1856.5 5207.47 1847.38 5216.59 1838.29 5225.73 1833.74 5238.09 1833.74 5253.66 1833.74 5269.77 1838.1 5282.54 1846.83 5291.97 1855.55 5301.43 1867.33 5306.16 1882.14 5306.16 1895.44 5306.16 1905.95 5301.98 1913.67 5293.62 1921.4 5285.26 1925.26 5273.89 1925.26 5259.52M1908.83 5263.92C1908.71 5272.48 1906.2 5279.31 1901.32 5284.42 1896.45 5289.53 1890.01 5292.08 1881.99 5292.08 1872.9 5292.08 1865.62 5289.61 1860.14 5284.68 1854.69 5279.74 1851.55 5272.79 1850.71 5263.81L1908.83 5263.92"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M2049.88 5341.36C2041.93 5327.94 2036.03 5314.67 2032.17 5301.54 2028.33 5288.44 2026.41 5275.14 2026.41 5261.65 2026.41 5248.18 2028.36 5234.84 2032.24 5221.61 2036.13 5208.41 2042.01 5195.14 2049.88 5181.79H2035.65C2027.07 5195.5 2020.64 5208.97 2016.37 5222.2 2012.11 5235.45 2009.99 5248.6 2009.99 5261.65 2009.99 5274.65 2012.1 5287.74 2016.33 5300.92 2020.56 5314.12 2027 5327.6 2035.65 5341.36H2049.88"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M2099.56 5320.24V5219.34H2120.68C2138.52 5219.34 2151.59 5223.39 2159.88 5231.51 2168.16 5239.63 2172.3 5252.43 2172.3 5269.94 2172.3 5287.32 2168.16 5300.04 2159.88 5308.11 2151.59 5316.2 2138.52 5320.24 2120.68 5320.24H2099.56M2080.79 5334.32H2118C2143.18 5334.32 2161.65 5329.12 2173.41 5318.7 2185.19 5308.31 2191.08 5292.06 2191.08 5269.94 2191.08 5247.67 2185.16 5231.31 2173.33 5220.88 2161.5 5210.46 2143.06 5205.26 2118 5205.26H2080.79V5334.32"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M2220.26 5334.32H2247.9L2280.06 5247.35 2312.4 5334.32H2339.94V5205.26H2321.16V5318.74L2288.68 5231.07H2271.52L2239.03 5318.74V5205.26H2220.26V5334.32"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M2374.38 5341.36H2388.6C2397.18 5327.6 2403.6 5314.12 2407.85 5300.92 2412.13 5287.74 2414.27 5274.65 2414.27 5261.65 2414.27 5248.6 2412.13 5235.45 2407.85 5222.2 2403.6 5208.97 2397.18 5195.5 2388.6 5181.79H2374.38C2382.25 5195.14 2388.13 5208.41 2392.01 5221.61 2395.9 5234.84 2397.84 5248.18 2397.84 5261.65 2397.84 5275.14 2395.9 5288.44 2392.01 5301.54 2388.13 5314.67 2382.25 5327.94 2374.38 5341.36"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M5174.9 6419.31H8210.04V7633.36O5H5174.9Z" fill="#ffffff" fill-rule="evenodd"/> <path transform="matrix(.1,0,0,-.1,0,796)" stroke-width="2.8346" stroke-linecap="butt" stroke-miterlimit="10" stroke-linejoin="miter" fill="none" stroke="#ffffff" d="M5174.9 6419.31H8210.04V7633.36O5H5174.9Z"/> <path transform="matrix(.1,0,0,-.1,0,796)" stroke-width="30.3515" stroke-linecap="butt" stroke-miterlimit="10" stroke-linejoin="miter" fill="none" stroke="#000000" d="M5174.9 6419.31H8210.04V7633.36O5H5174.9Z"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M5975.1 7307.12V7206.21H5996.22C6014.07 7206.21 6027.13 7210.27 6035.42 7218.39 6043.7 7226.5 6047.85 7239.31 6047.85 7256.81 6047.85 7274.19 6043.7 7286.91 6035.42 7294.98 6027.13 7303.07 6014.07 7307.12 5996.22 7307.12H5975.1M5956.33 7321.2H5993.54C6018.72 7321.2 6037.19 7315.99 6048.95 7305.58 6060.73 7295.19 6066.62 7278.93 6066.62 7256.81 6066.62 7234.54 6060.7 7218.19 6048.88 7207.75 6037.04 7197.34 6018.6 7192.13 5993.54 7192.13H5956.33V7321.2"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6180.28 7246.4V7239.07H6105.19C6105.89 7227.6 6109.24 7218.86 6115.23 7212.85 6121.22 7206.86 6129.56

7203.87 6140.24 7203.87 6146.45 7203.87 6152.46 7204.65 6158.28 7206.21 6164.1 7207.78 6169.87 7210.13 6175.59 7213.25V7199.17C6169.82 7196.12 6163.9 7193.78 6157.84 7192.17 6151.8 7190.58 6145.68 7189.79 6139.47 7189.79 6123.88 7189.79 6111.52 7194.34 6102.4 7203.46 6093.3 7212.61 6088.76 7224.96 6088.76 7240.53 6088.76 7256.64 6093.12 7269.41 6101.85 7278.85 6110.57 7288.31 6122.35 7293.04 6137.16 7293.04 6150.46 7293.04 6160.97 7288.86 6168.69 7280.5 6176.42 7272.14 6180.28 7260.77 6180.28 7246.4M6163.85 7250.8C6163.73 7259.36 6161.22 7266.19 6156.34 7271.3 6151.47 7276.41 6145.03 7278.96 6137.01 7278.96 6127.92 7278.96 6120.63 7276.49 6115.16 7271.55 6109.71 7266.61 6106.57 7259.66 6105.73 7250.69L6163.85 7250.8"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6278.06 7241.41C6278.06 7253.12 6275.65 7262.3 6270.84 7268.95 6266.05 7275.62 6259.46 7278.96 6251.07 7278.96 6242.69 7278.96 6236.09 7275.62 6231.27 7268.95 6226.48 7262.3 6224.09 7253.12 6224.09 7241.41 6224.09 7229.7 6226.48 7220.51 6231.27 7213.84 6236.09 7207.19 6242.69 7203.87 6251.07 7203.87 6259.46 7203.87 6266.05 7207.19 6270.84 7213.84 6275.65 7220.51 6278.06 7229.7 6278.06 7241.41M6224.09 7276.61C6227.41 7282.16 6231.62 7286.28 6236.7 7288.97 6241.79 7291.68 6247.86 7293.04 6254.92 7293.04 6266.66 7293.04 6276.18 7288.3 6283.48 7278.81 6290.82 7269.33 6294.48 7256.86 6294.48 7241.41 6294.48 7225.96 6290.82 7213.5 6283.48 7204.01 6276.18 7194.53 6266.66 7189.79 6254.92 7189.79 6247.86 7189.79 6241.79 7191.13 6236.7 7193.82 6231.62 7196.53 6227.41 7200.66 6224.09 7206.21V7192.13H6207.66V7328.24H6224.09V7276.61"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6323.06 7230.27V7290.69H6339.49V7230.89C6339.49 7221.89 6341.27 7215.14 6344.84 7210.61 6348.41 7206.11 6353.78 7203.87 6360.94 7203.87 6369.52 7203.87 6376.3 7206.57 6381.29 7211.97 6386.27 7217.39 6388.77 7224.78 6388.77 7234.12V7290.69H6405.2V7192.13H6388.77V7206.21C6384.88 7200.66 6380.36 7196.53 6375.2 7193.82 6370.07 7191.13 6364.11 7189.79 6357.31 7189.79 6346.09 7189.79 6337.57 7193.22 6331.75 7200.09 6325.96 7206.98 6323.06 7217.04 6323.06 7230.27M6363.62 7293.04V7293.04"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6501.11 7242.55C6501.11 7254.11 6498.74 7263.07 6494 7269.43 6489.28 7275.78 6482.64 7278.96 6474.09 7278.96 6465.59 7278.96 6458.96 7275.78 6454.22 7269.43 6449.5 7263.07 6447.14 7254.11 6447.14 7242.55 6447.14 7231.04 6449.5 7222.1 6454.22 7215.75 6458.96 7209.39 6465.59 7206.21 6474.09 7206.21 6482.64 7206.21 6489.28 7209.39 6494 7215.75 6498.74 7222.1 6501.11 7231.04 6501.11 7242.55M6517.54 7204.86C6517.54 7187.89 6513.86 7175.28 6506.5 7167.02 6499.17 7158.73 6487.94 7154.59 6472.81 7154.59 6467.18 7154.59 6461.89 7154.98 6456.93 7155.76 6451.97 7156.52 6447.14 7157.69 6442.45 7159.28V7175.71C6447.09 7173.31 6451.68 7171.54 6456.2 7170.39 6460.75 7169.24 6465.36 7168.67 6470.06 7168.67 6480.45 7168.67 6488.22 7171.42 6493.38 7176.92 6498.54 7182.42 6501.11 7190.71 6501.11 7201.81V7208.56C6497.84 7203.06 6493.64 7198.94 6488.54 7196.2 6483.43 7193.49 6477.3 7192.13 6470.17 7192.13 6458.34 7192.13 6448.8 7196.73 6441.57 7205.92 6434.33 7215.14 6430.71 7227.34 6430.71 7242.55 6430.71 7257.8 6434.33 7270.02 6441.57 7279.21 6448.8 7288.43 6458.34 7293.04 6470.17 7293.04 6477.3 7293.04 6483.43 7291.67 6488.54 7288.93 6493.64 7286.22 6497.84 7282.11 6501.11 7276.61V7290.69H6517.54V7204.86"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6591.73 7321.2H6704.37V7307.12H6657.43V7192.13H6638.66V7307.12H6591.73V7321.2"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6750.36 7276.61C6748.6 7277.42 6746.7 7278 6744.64 7278.37 6742.59 7278.76 6740.32 7278.96 6737.82 7278.96 6729.02 7278.96 6722.27 7276.01 6717.55 7270.12 6712.83 7264.25 6710.47 7255.82 6710.47 7244.82V7192.13H6694.04V7290.69H6710.47V7276.61C6713.75 7282.16 6718.01 7286.28 6723.27 7288.97 6728.52 7291.68 6734.9 7293.04 6742.41 7293.04 6743.48 7293.04 6744.67 7293.04 6745.96 7293.04 6747.26 7293.04 6748.7 7293.04 6750.29 7293.04L6750.36 7276.61"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6811.43 7241.41C6798.77 7241.41 6789.99 7239.96 6785.1 7237.05 6780.21 7234.14 6777.77 7229.17 6777.77 7222.13 6777.77 7216.53 6779.6 7212.08 6783.27 7208.78 6786.93 7205.5 6791.92 7203.87 6798.23 7203.87 6806.93 7203.87 6813.91 7206.95 6819.16 7213.11 6824.42 7219.29 6827.05 7227.5 6827.05 7237.75V7241.41H6811.43M6843.47 7247.61V7192.13H6827.05V7206.21C6823.43 7200.59 6818.92 7196.45 6813.52 7193.78 6808.12 7191.12 6801.5 7189.79 6793.68 7189.79 6783.78 7189.79 6775.91 7192.55 6770.07 7198.07 6764.25

7203.62 6761.34 7211.04 6761.34 7220.33 6761.34 7231.18 6765 7239.36 6772.3 7244.86 6779.61 7250.38 6790.53 7253.14 6805.05 7253.14H6827.05V7254.91C6827.05 7262.56 6824.64 7268.47 6819.82 7272.65 6815.04 7276.86 6808.3 7278.96 6799.62 7278.96 6794.1 7278.96 6788.72 7278.37 6783.49 7277.2 6778.26 7276.03 6773.22 7274.27 6768.38 7271.92V7286C6774.22 7288.34 6779.89 7290.11 6785.39 7291.28 6790.92 7292.45 6796.29 7293.04 6801.49 7293.04 6815.57 7293.04 6826.08 7289.27 6833.02 7281.75 6839.99 7274.24 6843.47 7262.86 6843.47 7247.61"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6961.67 7252.48V7192.13H6945.24V7251.97C6945.24 7260.99 6943.45 7267.74 6939.85 7272.21 6936.28 7276.71 6930.92 7278.96 6923.75 7278.96 6915.15 7278.96 6908.37 7276.25 6903.41 7270.82 6898.44 7265.42 6895.96 7258.04 6895.96 7248.67V7192.13H6879.54V7290.69H6895.96V7276.61C6899.8 7282.11 6904.32 7286.22 6909.53 7288.93 6914.73 7291.67 6920.73 7293.04 6927.53 7293.04 6938.78 7293.04 6947.27 7289.61 6953.02 7282.73 6958.79 7275.89 6961.67 7265.81 6961.67 7252.48"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M7055.24 7286V7271.92C7050.84 7274.27 7046.27 7276.03 7041.53 7277.2 7036.79 7278.37 7031.87 7278.96 7026.79 7278.96 7019.06 7278.96 7013.27 7277.75 7009.41 7275.33 7005.54 7272.91 7003.61 7269.28 7003.61 7264.44 7003.61 7260.75 7004.96 7257.85 7007.64 7255.75 7010.36 7253.65 7015.81 7251.64 7024 7249.73L7029.24 7248.53C7040.34 7246.08 7048.22 7242.62 7052.89 7238.15 7057.59 7233.68 7059.93 7227.43 7059.93 7219.41 7059.93 7210.3 7056.44 7203.07 7049.45 7197.74 7042.45 7192.44 7032.84 7189.79 7020.59 7189.79 7015.48 7189.79 7010.16 7190.38 7004.64 7191.55 6999.12 7192.72 6993.3 7194.48 6987.19 7196.83V7213.25C6992.93 7210.13 6998.59 7207.78 7004.16 7206.21 7009.74 7204.65 7015.25 7203.87 7020.7 7203.87 7028.03 7203.87 7033.67 7205.15 7037.6 7207.71 7041.54 7210.31 7043.51 7213.94 7043.51 7218.61 7043.51 7222.96 7042.11 7226.28 7039.32 7228.58 7036.54 7230.9 7030.42 7233.14 7020.96 7235.29L7015.64 7236.57C7005.59 7238.68 6998.33 7241.9 6993.86 7246.25 6989.41 7250.61 6987.19 7256.57 6987.19 7264.14 6987.19 7273.36 6990.41 7280.47 6996.87 7285.48 7003.32 7290.52 7012.47 7293.04 7024.33 7293.04 7030.2 7293.04 7035.72 7292.45 7040.9 7291.28 7046.09 7290.11 7050.86 7288.34 7055.24 7286"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M7104.81 7206.21V7154.59H7088.39V7290.69H7104.81V7276.61C7108.14 7282.16 7112.34 7286.28 7117.43 7288.97 7122.51 7291.68 7128.59 7293.04 7135.65 7293.04 7147.38 7293.04 7156.91 7288.3 7164.21 7278.81 7171.55 7269.33 7175.21 7256.86 7175.21 7241.41 7175.21 7225.96 7171.55 7213.5 7164.21 7204.01 7156.91 7194.53 7147.38 7189.79 7135.65 7189.79 7128.59 7189.79 7122.51 7191.13 7117.43 7193.82 7112.34 7196.53 7108.14 7200.66 7104.81 7206.21M7158.79 7241.41C7158.79 7253.12 7156.38 7262.3 7151.56 7268.95 7146.77 7275.62 7140.18 7278.96 7131.8 7278.96 7123.42 7278.96 7116.82 7275.62 7112 7268.95 7107.21 7262.3 7104.81 7253.12 7104.81 7241.41 7104.81 7229.7 7107.21 7220.51 7112 7213.84 7116.82 7207.19 7123.42 7203.87 7131.8 7203.87 7140.18 7203.87 7146.77 7207.19 7151.56 7213.84 7156.38 7220.51 7158.79 7229.7 7158.79 7241.41"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M7241.37 7278.96C7232.67 7278.96 7225.79 7275.61 7220.73 7268.91 7215.7 7262.21 7213.18 7253.05 7213.18 7241.41 7213.18 7229.78 7215.68 7220.61 7220.69 7213.91 7225.73 7207.21 7232.62 7203.87 7241.37 7203.87 7250.03 7203.87 7256.87 7207.23 7261.91 7213.95 7266.96 7220.67 7269.5 7229.82 7269.5 7241.41 7269.5 7252.95 7266.96 7262.09 7261.91 7268.84 7256.87 7275.59 7250.03 7278.96 7241.37 7278.96M7241.34 7293.04C7255.22 7293.04 7266.12 7288.47 7274.04 7279.32 7281.96 7270.21 7285.92 7257.57 7285.92 7241.41 7285.92 7225.3 7281.96 7212.67 7274.04 7203.5 7266.12 7194.36 7255.22 7189.79 7241.34 7189.79 7227.4 7189.79 7216.49 7194.36 7208.59 7203.5 7200.7 7212.67 7196.75 7225.3 7196.75 7241.41 7196.75 7257.57 7200.7 7270.21 7208.59 7279.32 7216.49 7288.47 7227.4 7293.04 7241.34 7293.04"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M7371.26 7276.61C7369.5 7277.42 7367.59 7278 7365.54 7278.37 7363.49 7278.76 7361.21 7278.96 7358.72 7278.96 7349.92 7278.96 7343.16 7276.01 7338.45 7270.12 7333.73 7264.25 7331.37 7255.82 7331.37 7244.82V7192.13H7314.94V7290.69H7331.37V7276.61C7334.64 7282.16 7338.91 7286.28 7344.16 7288.97 7349.42 7291.68 7355.8 7293.04 7363.3 7293.04 7364.38 7293.04 7365.57 7293.04 7366.86 7293.04 7368.16 7293.04 7369.6 7293.04 7371.19 7293.04L7371.26 7276.61"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M7405.71 7318.85V7290.69H7438.56V7278.96H7405.71V7225.21C7405.71 7217.14 7406.78 7211.96 7408.93

7209.66 7411.11 7207.36 7415.52 7206.21 7422.17 7206.21H7438.56V7192.13H7422.17C7409.73 7192.13 7401.14 7194.5 7396.39 7199.25 7391.65 7204.01 7389.28 7212.67 7389.28 7225.21V7278.96H7377.55V7290.69H7389.28V7318.85H7405.71"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6270.61 7094.43H6289.39V7042.8H6350.4V7094.43H6369.17V6965.36H6350.4V7028.72H6289.39V6965.36H6270.61V7094.43"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6450.3 7014.64C6437.64 7014.64 6428.86 7013.19 6423.98 7010.28 6419.09 7007.37 6416.64 7002.4 6416.64 6995.36 6416.64 6989.76 6418.48 6985.31 6422.14 6982.01 6425.81 6978.73 6430.79 6977.1 6437.1 6977.1 6445.8 6977.1 6452.78 6980.18 6458.04 6986.34 6463.29 6992.52 6465.92 7000.73 6465.92 7010.98V7014.64H6450.3M6482.35 7020.84V6965.36H6465.92V6979.45C6462.3 6973.82 6457.79 6969.68 6452.39 6967.02 6446.99 6964.35 6440.38 6963.02 6432.55 6963.02 6422.66 6963.02 6414.79 6965.78 6408.94 6971.3 6403.13 6976.86 6400.21 6984.27 6400.21 6993.56 6400.21 7004.41 6403.87 7012.59 6411.18 7018.09 6418.49 7023.62 6429.4 7026.38 6443.92 7026.38H6465.92V7028.14C6465.92 7035.79 6463.51 7041.7 6458.7 7045.88 6453.91 7050.09 6447.17 7052.19 6438.5 7052.19 6432.97 7052.19 6427.59 7051.61 6422.36 7050.43 6417.13 7049.26 6412.09 7047.5 6407.25 7045.15V7059.23C6413.1 7061.58 6418.77 7063.34 6424.27 7064.51 6429.79 7065.68 6435.16 7066.27 6440.36 7066.27 6454.45 7066.27 6464.96 7062.51 6471.9 7054.98 6478.86 7047.47 6482.35 7036.09 6482.35 7020.84"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6574.73 7049.84C6572.97 7050.65 6571.06 7051.24 6569.01 7051.61 6566.95 7052 6564.68 7052.19 6562.19 7052.19 6553.39 7052.19 6546.63 7049.25 6541.91 7043.36 6537.19 7037.49 6534.84 7029.05 6534.84 7018.05V6965.36H6518.41V7063.93H6534.84V7049.84C6538.11 7055.39 6542.38 7059.51 6547.63 7062.2 6552.89 7064.91 6559.27 7066.27 6566.77 7066.27 6567.85 7066.27 6569.03 7066.27 6570.33 7066.27 6571.62 7066.27 6573.07 7066.27 6574.65 7066.27L6574.73 7049.84"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6653.27 7049.84V7101.47H6669.7V6965.36H6653.27V6979.45C6649.95 6973.89 6645.75 6969.77 6640.66 6967.05 6635.58 6964.36 6629.46 6963.02 6622.33 6963.02 6610.67 6963.02 6601.17 6967.76 6593.84 6977.25 6586.53 6986.73 6582.88 6999.2 6582.88 7014.64 6582.88 7030.09 6586.53 7042.56 6593.84 7052.04 6601.17 7061.53 6610.67 7066.27 6622.33 7066.27 6629.46 7066.27 6635.58 7064.91 6640.66 7062.2 6645.75 7059.51 6649.95 7055.39 6653.27 7049.84M6599.3 7014.64C6599.3 7002.94 6601.7 6993.75 6606.49 6987.07 6611.28 6980.42 6617.87 6977.1 6626.25 6977.1 6634.64 6977.1 6641.23 6980.42 6646.05 6987.07 6650.87 6993.75 6653.27 7002.94 6653.27 7014.64 6653.27 7026.35 6650.87 7035.53 6646.05 7042.18 6641.23 7048.86 6634.64 7052.19 6626.25 7052.19 6617.87 7052.19 6611.28 7048.86 6606.49 7042.18 6601.7 7035.53 6599.3 7026.35 6599.3 7014.64"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6695.93 7063.93H6712.03L6732.12 6987 6752.14 7063.93H6771.13L6791.23 6987 6811.25 7063.93H6827.34L6801.71 6965.36H6782.75L6761.67 7046.14 6740.52 6965.36H6721.56L6695.93 7063.93"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6896.33 7014.64C6883.67 7014.64 6874.89 7013.19 6870 7010.28 6865.11 7007.37 6862.67 7002.4 6862.67 6995.36 6862.67 6989.76 6864.5 6985.31 6868.17 6982.01 6871.84 6978.73 6876.82 6977.1 6883.13 6977.1 6891.83 6977.1 6898.81 6980.18 6904.07 6986.34 6909.32 6992.52 6911.95 7000.73 6911.95 7010.98V7014.64H6896.33M6928.38 7020.84V6965.36H6911.95V6979.45C6908.33 6973.82 6903.82 6969.68 6898.42 6967.02 6893.02 6964.35 6886.41 6963.02 6878.58 6963.02 6868.68 6963.02 6860.81 6965.78 6854.97 6971.3 6849.15 6976.86 6846.25 6984.27 6846.25 6993.56 6846.25 7004.41 6849.9 7012.59 6857.21 7018.09 6864.52 7023.62 6875.43 7026.38 6889.95 7026.38H6911.95V7028.14C6911.95 7035.79 6909.54 7041.7 6904.73 7045.88 6899.94 7050.09 6893.2 7052.19 6884.52 7052.19 6879 7052.19 6873.62 7051.61 6868.39 7050.43 6863.16 7049.26 6858.13 7047.5 6853.29 7045.15V7059.23C6859.13 7061.58 6864.8 7063.34 6870.3 7064.51 6875.82 7065.68 6881.19 7066.27 6886.39 7066.27 6900.47 7066.27 6910.98 7062.51 6917.93 7054.98 6924.89 7047.47 6928.38 7036.09 6928.38 7020.84"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M7020.76 7049.84C7019 7050.65 7017.09 7051.24 7015.04 7051.61 7012.98 7052 7010.71 7052.19 7008.21 7052.19 6999.42 7052.19 6992.66 7049.25 6987.94 7043.36 6983.22 7037.49 6980.86

7029.05 6980.86 7018.05V6965.36H6964.44V7063.93H6980.86V7049.84C6984.14 7055.39 6988.41 7059.51 6993.66 7062.2 6998.91 7064.91 7005.3 7066.27 7012.8 7066.27 7013.88 7066.27 7015.06 7066.27 7016.36 7066.27 7017.65 7066.27 7019.09 7066.27 7020.68 7066.27L7020.76 7049.84"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M7119 7019.63V7012.3H7043.91C7044.62 7000.83 7047.97 6992.09 7053.96 6986.08 7059.95 6980.09 7068.29 6977.1 7078.96 6977.1 7085.18 6977.1 7091.19 6977.88 7097 6979.45 7102.82 6981.01 7108.59 6983.36 7114.31 6986.48V6972.41C7108.54 6969.35 7102.63 6967.02 7096.57 6965.4 7090.53 6963.81 7084.41 6963.02 7078.2 6963.02 7062.6 6963.02 7050.24 6967.58 7041.13 6976.7 7032.03 6985.84 7027.49 6998.2 7027.49 7013.77 7027.49 7029.88 7031.85 7042.64 7040.58 7052.08 7049.3 7061.54 7061.07 7066.27 7075.89 7066.27 7089.18 7066.27 7099.7 7062.09 7107.42 7053.73 7115.14 7045.37 7119 7034 7119 7019.63M7102.58 7024.03C7102.46 7032.59 7099.95 7039.42 7095.06 7044.53 7090.2 7049.64 7083.76 7052.19 7075.74 7052.19 7066.64 7052.19 7059.36 7049.72 7053.89 7044.79 7048.43 7039.85 7045.29 7032.89 7044.46 7023.92L7102.58 7024.03"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M5694.69 6874.7C5686.75 6861.28 5680.84 6848.01 5676.98 6834.88 5673.14 6821.78 5671.22 6808.48 5671.22 6794.99 5671.22 6781.52 5673.17 6768.18 5677.05 6754.95 5680.94 6741.75 5686.82 6728.48 5694.69 6715.13H5680.46C5671.88 6728.84 5665.45 6742.31 5661.18 6755.54 5656.92 6768.79 5654.8 6781.94 5654.8 6794.99 5654.8 6807.99 5656.91 6821.08 5661.14 6834.26 5665.37 6847.46 5671.81 6860.94 5680.46 6874.7H5694.69"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M5810.07 6792.86V6785.53H5734.98C5735.69 6774.07 5739.04 6765.33 5745.03 6759.31 5751.02 6753.32 5759.35 6750.33 5770.04 6750.33 5776.25 6750.33 5782.26 6751.11 5788.07 6752.68 5793.89 6754.24 5799.66 6756.59 5805.38 6759.71V6745.64C5799.61 6742.58 5793.7 6740.25 5787.64 6738.63 5781.6 6737.04 5775.47 6736.25 5769.27 6736.25 5753.67 6736.25 5741.31 6740.81 5732.2 6749.93 5723.1 6759.07 5718.55 6771.43 5718.55 6787 5718.55 6803.11 5722.92 6815.88 5731.64 6825.31 5740.37 6834.77 5752.14 6839.5 5766.96 6839.5 5780.25 6839.5 5790.77 6835.32 5798.49 6826.96 5806.21 6818.6 5810.07 6807.23 5810.07 6792.86M5793.65 6797.26C5793.53 6805.82 5791.02 6812.65 5786.13 6817.76 5781.27 6822.87 5774.83 6825.42 5766.81 6825.42 5757.71 6825.42 5750.43 6822.95 5744.96 6818.02 5739.5 6813.08 5736.36 6806.13 5735.53 6797.15L5793.65 6797.26"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M5900.82 6789.01C5900.82 6800.57 5898.45 6809.54 5893.7 6815.89 5888.98 6822.25 5882.35 6825.42 5873.79 6825.42 5865.29 6825.42 5858.66 6822.25 5853.92 6815.89 5849.2 6809.54 5846.84 6800.57 5846.84 6789.01 5846.84 6777.5 5849.2 6768.57 5853.92 6762.21 5858.66 6755.86 5865.29 6752.68 5873.79 6752.68 5882.35 6752.68 5888.98 6755.86 5893.7 6762.21 5898.45 6768.57 5900.82 6777.5 5900.82 6789.01M5917.24 6751.32C5917.24 6734.36 5913.56 6721.74 5906.21 6713.48 5898.87 6705.2 5887.64 6701.05 5872.51 6701.05 5866.89 6701.05 5861.59 6701.44 5856.63 6702.22 5851.67 6702.98 5846.84 6704.16 5842.15 6705.74V6722.17C5846.79 6719.77 5851.38 6718 5855.9 6716.86 5860.45 6715.7 5865.07 6715.13 5869.76 6715.13 5880.15 6715.13 5887.92 6717.88 5893.08 6723.38 5898.24 6728.88 5900.82 6737.18 5900.82 6748.28V6755.02C5897.54 6749.52 5893.35 6745.41 5888.24 6742.67 5883.13 6739.95 5877.01 6738.6 5869.87 6738.6 5858.04 6738.6 5848.5 6743.19 5841.27 6752.38 5834.04 6761.6 5830.42 6773.81 5830.42 6789.01 5830.42 6804.27 5834.04 6816.49 5841.27 6825.68 5848.5 6834.89 5858.04 6839.5 5869.87 6839.5 5877.01 6839.5 5883.13 6838.13 5888.24 6835.39 5893.35 6832.68 5897.54 6828.57 5900.82 6823.07V6837.16H5917.24V6751.32"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M5957.55 6759.71H5976.32V6738.6H5957.55V6759.71"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6067.55 6867.66H6086.32V6747.29C6086.32 6731.91 6083.3 6720.75 6077.27 6713.81 6071.25 6706.87 6061.57 6703.4 6048.23 6703.4H6041.74V6717.48H6047.02C6054.55 6717.48 6059.85 6719.65 6062.93 6724 6066.01 6728.33 6067.55 6736.09 6067.55 6747.29V6867.66"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6102.59 6867.66H6215.22V6853.58H6168.29V6738.6H6149.52V6853.58H6102.59V6867.66"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6263.4 6850.32 6239.42 6785.53H6287.45L6263.4 6850.32M6253.43 6867.66H6273.45L6323.24 6738.6H6304.87L6292.95 6771.45H6234.11L6222.19 6738.6H6203.56L6253.43 6867.66"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6425.53 6757.08V6792.57H6397.37V6806.65H6444.3V6750.51C6437.56 6745.82 6430.13 6742.26 6422.01


6739.84 6413.9 6737.45 6405.23 6736.25 6396.02 6736.25 6375.88 6736.25 6360.11 6742.13 6348.71 6753.89 6337.35 6765.67 6331.67 6782.07 6331.67 6803.09 6331.67 6824.16 6337.4 6840.58 6348.86 6852.34 6360.35 6864.12 6376.28 6870.01 6396.64 6870.01 6405.12 6870.01 6413.18 6869.02 6420.8 6867.04 6428.45 6865.06 6435.5 6862.14 6441.96 6858.28V6839.5C6435.46 6844.95 6428.55 6849.05 6421.24 6851.79 6413.93 6854.55 6406.26 6855.93 6398.21 6855.93 6382.3 6855.93 6370.36 6851.5 6362.39 6842.66 6354.43 6833.81 6350.44 6820.62 6350.44 6803.09 6350.44 6785.62 6354.3 6772.45 6362.03 6763.6 6369.75 6754.75 6381.31 6750.33 6396.71 6750.33 6402.73 6750.33 6408.09 6750.88 6412.81 6751.98 6417.53 6753.08 6421.77 6754.78 6425.53 6757.08"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6599.59 6823.07V6874.7H6616.02V6738.6H6599.59V6752.68C6596.27 6747.13 6592.06 6743 6586.98 6740.29 6581.89 6737.59 6575.78 6736.25 6568.64 6736.25 6556.98 6736.25 6547.49 6740.99 6540.16 6750.48 6532.85 6759.96 6529.19 6772.43 6529.19 6787.88 6529.19 6803.32 6532.85 6815.79 6540.16 6825.28 6547.49 6834.76 6556.98 6839.5 6568.64 6839.5 6575.78 6839.5 6581.89 6838.14 6586.98 6835.43 6592.06 6832.74 6596.27 6828.63 6599.59 6823.07M6545.62 6787.88C6545.62 6776.17 6548.02 6766.98 6552.8 6760.3 6557.6 6753.66 6564.18 6750.33 6572.57 6750.33 6580.95 6750.33 6587.55 6753.66 6592.37 6760.3 6597.18 6766.98 6599.59 6776.17 6599.59 6787.88 6599.59 6799.59 6597.18 6808.77 6592.37 6815.41 6587.55 6822.09 6580.95 6825.42 6572.57 6825.42 6564.18 6825.42 6557.6 6822.09 6552.8 6815.41 6548.02 6808.77 6545.62 6799.59 6545.62 6787.88"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6736.11 6792.86V6785.53H6661.02C6661.73 6774.07 6665.08 6765.33 6671.07 6759.31 6677.05 6753.32 6685.39 6750.33 6696.07 6750.33 6702.28 6750.33 6708.3 6751.11 6714.11 6752.68 6719.93 6754.24 6725.7 6756.59 6731.42 6759.71V6745.64C6725.65 6742.58 6719.73 6740.25 6713.67 6738.63 6707.64 6737.04 6701.51 6736.25 6695.3 6736.25 6679.71 6736.25 6667.35 6740.81 6658.23 6749.93 6649.14 6759.07 6644.59 6771.43 6644.59 6787 6644.59 6803.11 6648.96 6815.88 6657.68 6825.31 6666.41 6834.77 6678.18 6839.5 6692.99 6839.5 6706.29 6839.5 6716.8 6835.32 6724.53 6826.96 6732.25 6818.6 6736.11 6807.23 6736.11 6792.86M6719.69 6797.26C6719.56 6805.82 6717.06 6812.65 6712.17 6817.76 6707.3 6822.87 6700.86 6825.42 6692.85 6825.42 6683.75 6825.42 6676.47 6822.95 6670.99 6818.02 6665.54 6813.08 6662.4 6806.13 6661.57 6797.15L6719.69 6797.26"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6833.89 6787.88C6833.89 6799.59 6831.48 6808.77 6826.67 6815.41 6821.88 6822.09 6815.29 6825.42 6806.91 6825.42 6798.52 6825.42 6791.92 6822.09 6787.11 6815.41 6782.32 6808.77 6779.92 6799.59 6779.92 6787.88 6779.92 6776.17 6782.32 6766.98 6787.11 6760.3 6791.92 6753.66 6798.52 6750.33 6806.91 6750.33 6815.29 6750.33 6821.88 6753.66 6826.67 6760.3 6831.48 6766.98 6833.89 6776.17 6833.89 6787.88M6779.92 6823.07C6783.25 6828.63 6787.45 6832.74 6792.54 6835.43 6797.62 6838.14 6803.69 6839.5 6810.76 6839.5 6822.49 6839.5 6832.01 6834.76 6839.32 6825.28 6846.65 6815.79 6850.32 6803.32 6850.32 6787.88 6850.32 6772.43 6846.65 6759.96 6839.32 6750.48 6832.01 6740.99 6822.49 6736.25 6810.76 6736.25 6803.69 6736.25 6797.62 6737.59 6792.54 6740.29 6787.45 6743 6783.25 6747.13 6779.92 6752.68V6738.6H6763.5V6874.7H6779.92V6823.07"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6878.9 6776.73V6837.16H6895.32V6777.35C6895.32 6768.36 6897.11 6761.6 6900.68 6757.08 6904.25 6752.58 6909.61 6750.33 6916.77 6750.33 6925.35 6750.33 6932.14 6753.03 6937.13 6758.43 6942.11 6763.86 6944.6 6771.24 6944.6 6780.58V6837.16H6961.03V6738.6H6944.6V6752.68C6940.71 6747.13 6936.2 6743 6931.04 6740.29 6925.9 6737.59 6919.94 6736.25 6913.14 6736.25 6901.92 6736.25 6893.41 6739.68 6887.59 6746.55 6881.79 6753.45 6878.9 6763.5 6878.9 6776.73M6919.45 6839.5V6839.5"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M7056.95 6789.01C7056.95 6800.57 7054.58 6809.54 7049.84 6815.89 7045.12 6822.25 7038.48 6825.42 7029.93 6825.42 7021.42 6825.42 7014.79 6822.25 7010.05 6815.89 7005.34 6809.54 7002.98 6800.57 7002.98 6789.01 7002.98 6777.5 7005.34 6768.57 7010.05 6762.21 7014.79 6755.86 7021.42 6752.68 7029.93 6752.68 7038.48 6752.68 7045.12 6755.86 7049.84 6762.21 7054.58 6768.57 7056.95 6777.5 7056.95 6789.01M7073.38 6751.32C7073.38 6734.36 7069.7 6721.74 7062.34 6713.48 7055 6705.2 7043.77 6701.05 7028.64 6701.05 7023.02 6701.05 7017.73 6701.44 7012.77 6702.22 7007.8 6702.98 7002.98 6704.16 6998.28

6705.74V6722.17C7002.93 6719.77 7007.51 6718 7012.03 6716.86 7016.58 6715.7 7021.2 6715.13 7025.89 6715.13 7036.28 6715.13 7044.05 6717.88 7049.21 6723.38 7054.37 6728.88 7056.95 6737.18 7056.95 6748.28V6755.02C7053.67 6749.52 7049.48 6745.41 7044.37 6742.67 7039.26 6739.95 7033.14 6738.6 7026 6738.6 7014.17 6738.6 7004.64 6743.19 6997.4 6752.38 6990.17 6761.6 6986.55 6773.81 6986.55 6789.01 6986.55 6804.27 6990.17 6816.49 6997.4 6825.68 7004.64 6834.89 7014.17 6839.5 7026 6839.5 7033.14 6839.5 7039.26 6838.13 7044.37 6835.39 7049.48 6832.68 7053.67 6828.57 7056.95 6823.07V6837.16H7073.38V6751.32"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M7182.76 6752.68V6701.05H7166.34V6837.16H7182.76V6823.07C7186.09 6828.63 7190.29 6832.74 7195.38 6835.43 7200.46 6838.14 7206.54 6839.5 7213.6 6839.5 7225.33 6839.5 7234.86 6834.76 7242.16 6825.28 7249.5 6815.79 7253.16 6803.32 7253.16 6787.88 7253.16 6772.43 7249.5 6759.96 7242.16 6750.48 7234.86 6740.99 7225.33 6736.25 7213.6 6736.25 7206.54 6736.25 7200.46 6737.59 7195.38 6740.29 7190.29 6743 7186.09 6747.13 7182.76 6752.68M7236.73 6787.88C7236.73 6799.59 7234.33 6808.77 7229.51 6815.41 7224.72 6822.09 7218.13 6825.42 7209.75 6825.42 7201.37 6825.42 7194.77 6822.09 7189.95 6815.41 7185.16 6808.77 7182.76 6799.59 7182.76 6787.88 7182.76 6776.17 7185.16 6766.98 7189.95 6760.3 7194.77 6753.66 7201.37 6750.33 7209.75 6750.33 7218.13 6750.33 7224.72 6753.66 7229.51 6760.3 7234.33 6766.98 7236.73 6776.17 7236.73 6787.88"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M7338.06 6823.07C7336.3 6823.88 7334.39 6824.47 7332.34 6824.84 7330.29 6825.23 7328.01 6825.42 7325.52 6825.42 7316.72 6825.42 7309.96 6822.48 7305.24 6816.59 7300.52 6810.72 7298.16 6802.29 7298.16 6791.29V6738.6H7281.74V6837.16H7298.16V6823.07C7301.44 6828.63 7305.71 6832.74 7310.96 6835.43 7316.22 6838.14 7322.6 6839.5 7330.1 6839.5 7331.18 6839.5 7332.36 6839.5 7333.66 6839.5 7334.95 6839.5 7336.39 6839.5 7337.98 6839.5L7338.06 6823.07"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M7389.41 6825.42C7380.71 6825.42 7373.83 6822.07 7368.77 6815.38 7363.73 6808.68 7361.21 6799.51 7361.21 6787.88 7361.21 6776.24 7363.72 6767.07 7368.73 6760.38 7373.77 6753.68 7380.66 6750.33 7389.41 6750.33 7398.06 6750.33 7404.91 6753.69 7409.95 6760.41 7415 6767.14 7417.54 6776.29 7417.54 6787.88 7417.54 6799.41 7415 6808.55 7409.95 6815.3 7404.91 6822.05 7398.06 6825.42 7389.41 6825.42M7389.38 6839.5C7403.26 6839.5 7414.16 6834.93 7422.08 6825.79 7430 6816.67 7433.96 6804.04 7433.96 6787.88 7433.96 6771.77 7430 6759.13 7422.08 6749.96 7414.16 6740.82 7403.26 6736.25 7389.38 6736.25 7375.44 6736.25 7364.53 6740.82 7356.63 6749.96 7348.73 6759.13 7344.79 6771.77 7344.79 6787.88 7344.79 6804.04 7348.73 6816.67 7356.63 6825.79 7364.53 6834.93 7375.44 6839.5 7389.38 6839.5"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M7533.38 6787.88C7533.38 6799.59 7530.97 6808.77 7526.16 6815.41 7521.37 6822.09 7514.78 6825.42 7506.39 6825.42 7498.01 6825.42 7491.41 6822.09 7486.59 6815.41 7481.8 6808.77 7479.41 6799.59 7479.41 6787.88 7479.41 6776.17 7481.8 6766.98 7486.59 6760.3 7491.41 6753.66 7498.01 6750.33 7506.39 6750.33 7514.78 6750.33 7521.37 6753.66 7526.16 6760.3 7530.97 6766.98 7533.38 6776.17 7533.38 6787.88M7479.41 6823.07C7482.73 6828.63 7486.94 6832.74 7492.02 6835.43 7497.11 6838.14 7503.18 6839.5 7510.24 6839.5 7521.98 6839.5 7531.5 6834.76 7538.8 6825.28 7546.14 6815.79 7549.8 6803.32 7549.8 6787.88 7549.8 6772.43 7546.14 6759.96 7538.8 6750.48 7531.5 6740.99 7521.98 6736.25 7510.24 6736.25 7503.18 6736.25 7497.11 6737.59 7492.02 6740.29 7486.94 6743 7482.73 6747.13 7479.41 6752.68V6738.6H7462.98V6874.7H7479.41V6823.07"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M7662.86 6792.86V6785.53H7587.77C7588.48 6774.07 7591.83 6765.33 7597.82 6759.31 7603.8 6753.32 7612.14 6750.33 7622.82 6750.33 7629.03 6750.33 7635.04 6751.11 7640.86 6752.68 7646.68 6754.24 7652.45 6756.59 7658.17 6759.71V6745.64C7652.4 6742.58 7646.48 6740.25 7640.42 6738.63 7634.38 6737.04 7628.26 6736.25 7622.05 6736.25 7606.46 6736.25 7594.1 6740.81 7584.98 6749.93 7575.89 6759.07 7571.34 6771.43 7571.34 6787 7571.34 6803.11 7575.71 6815.88 7584.43 6825.31 7593.16 6834.77 7604.93 6839.5 7619.74 6839.5 7633.04 6839.5 7643.55 6835.32 7651.28 6826.96 7659 6818.6 7662.86 6807.23 7662.86 6792.86M7646.43 6797.26C7646.31 6805.82 7643.81 6812.65 7638.92 6817.76 7634.05 6822.87 7627.61 6825.42 7619.6 6825.42 7610.5 6825.42 7603.22 6822.95 7597.74 6818.02 7592.29 6813.08 7589.15 6806.13 7588.32 6797.15L7646.43 6797.26"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M7687.9 6874.7H7702.12C7710.7 6860.94 7717.12 6847.46

7721.37 6834.26 7725.65 6821.08 7727.79 6807.99 7727.79 6794.99 7727.79 6781.94 7725.65 6768.79 7721.37 6755.54 7717.12 6742.31 7710.7 6728.84 7702.12 6715.13H7687.9C7695.77 6728.48 7701.64 6741.75 7705.53 6754.95 7709.42 6768.18 7711.36 6781.52 7711.36 6794.99 7711.36 6808.48 7709.42 6821.78 7705.53 6834.88 7701.64 6848.01 7695.77 6861.28 7687.9 6874.7"/> <path transform="matrix(.1,0,0,-.1,0,796)" stroke-width="30.3515" stroke-linecap="butt" stroke-miterlimit="10" stroke-linejoin="miter" fill="none" stroke="#000000" d="M6692.49 6271.56V5960.04"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6692.49 6385.38 6768.34 6233.62 6692.49 6271.56 6616.61 6233.62" fill-rule="evenodd"/> <path transform="matrix(.1,0,0,-.1,0,796)" stroke-width="2.8346" stroke-linecap="butt" stroke-miterlimit="10" stroke-linejoin="miter" fill="none" stroke="#000000" d="M6692.49 6385.38 6768.34 6233.62 6692.49 6271.56 6616.61 6233.62Z"/> <path transform="matrix(.1,0,0,-.1,0,796)" stroke-width="30.3515" stroke-linecap="butt" stroke-miterlimit="10" stroke-linejoin="miter" fill="none" stroke="#000000" d="M6692.49 6385.38 6768.34 6233.62 6692.49 6271.56 6616.61 6233.62Z"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6692.49 5846.21 6616.61 5997.97 6692.49 5960.04 6768.34 5997.97" fill-rule="evenodd"/> <path transform="matrix(.1,0,0,-.1,0,796)" stroke-width="2.8346" stroke-linecap="butt" stroke-miterlimit="10" stroke-linejoin="miter" fill="none" stroke="#000000" d="M6692.49 5846.21 6616.61 5997.97 6692.49 5960.04 6768.34 5997.97Z"/> <path transform="matrix(.1,0,0,-.1,0,796)" stroke-width="30.3515" stroke-linecap="butt" stroke-miterlimit="10" stroke-linejoin="miter" fill="none" stroke="#000000" d="M6692.49 5846.21 6616.61 5997.97 6692.49 5960.04 6768.34 5997.97Z"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M15.1758 6419.31H4871.396V7936.88H15.1758Z" fill="#ffffff" fill-rule="evenodd"/> <path transform="matrix(.1,0,0,-.1,0,796)" stroke-width="2.8346" stroke-linecap="butt" stroke-miterlimit="10" stroke-linejoin="miter" fill="none" stroke="#ffffff" d="M15.1758 6419.31H4871.396V7936.88H15.1758Z"/> <path transform="matrix(.1,0,0,-.1,0,796)" stroke-width="30.3515" stroke-linecap="butt" stroke-miterlimit="10" stroke-linejoin="miter" fill="none" stroke="#000000" d="M15.1758 6419.31H4871.396V7936.88H15.1758Z"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M353.887 7748.35V7647.45H375.008C392.852 7647.45 405.918 7651.5 414.203 7659.62 422.488 7667.73 426.633 7680.54 426.633 7698.04 426.633 7715.42 422.488 7728.14 414.203 7736.21 405.918 7744.3 392.852 7748.35 375.008 7748.35H353.887M335.113 7762.43H372.332C397.508 7762.43 415.977 7757.22 427.734 7746.81 439.516 7736.42 445.406 7720.16 445.406 7698.04 445.406 7675.77 439.492 7659.42 427.66 7648.98 415.828 7638.57 397.387 7633.36 372.332 7633.36H335.113V7762.43"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M559.066 7687.63V7680.3H483.973C484.68 7668.83 488.031 7660.09 494.02 7654.08 500.008 7648.09 508.344 7645.1 519.027 7645.1 525.234 7645.1 531.246 7645.88 537.066 7647.45 542.883 7649.01 548.652 7651.36 554.371 7654.48V7640.4C548.602 7637.35 542.688 7635.02 536.625 7633.4 530.586 7631.81 524.465 7631.02 518.254 7631.02 502.66 7631.02 490.305 7635.58 481.188 7644.7 472.094 7653.84 467.547 7666.19 467.547 7681.76 467.547 7697.87 471.91 7710.64 480.637 7720.08 489.363 7729.54 501.133 7734.27 515.945 7734.27 529.242 7734.27 539.754 7730.09 547.48 7721.73 555.203 7713.37 559.066 7702 559.066 7687.63M542.641 7692.03C542.516 7700.59 540.012 7707.42 535.121 7712.53 530.258 7717.64 523.816 7720.19 515.801 7720.19 506.707 7720.19 499.422 7717.72 493.945 7712.79 488.496 7707.84 485.355 7700.89 484.523 7691.92L542.641 7692.03"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M656.848 7682.64C656.848 7694.35 654.438 7703.53 649.621 7710.18 644.832 7716.85 638.242 7720.19 629.859 7720.19 621.477 7720.19 614.875 7716.85 610.059 7710.18 605.27 7703.53 602.875 7694.35 602.875 7682.64 602.875 7670.93 605.27 7661.74 610.059 7655.07 614.875 7648.42 621.477 7645.1 629.859 7645.1 638.242 7645.1 644.832 7648.42 649.621 7655.07 654.438 7661.74 656.848 7670.93 656.848 7682.64M602.875 7717.84C606.199 7723.39 610.402 7727.51 615.488 7730.2 620.57 7732.91 626.645 7734.27 633.711 7734.27 645.441 7734.27 654.965 7729.53 662.273 7720.04 669.605 7710.56 673.273 7698.09 673.273 7682.64 673.273 7667.2 669.605 7654.73 662.273 7645.24 654.965 7635.76 645.441 7631.02 633.711 7631.02 626.645 7631.02 620.57 7632.36 615.488 7635.05 610.402 7637.77 606.199 7641.89 602.875 7647.45V7633.36H586.445V7769.47H602.875V7717.84"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M701.848 7671.5V7731.92H718.273V7672.12C718.273 7663.13 720.059 7656.37 723.629 7651.84

727.199 7647.35 732.563 7645.1 739.727 7645.1 748.305 7645.1 755.09 7647.8 760.074 7653.2 765.063 7658.63 767.555 7666.01 767.555 7675.35V7731.92H783.98V7633.36H767.555V7647.45C763.668 7641.89 759.145 7637.77 753.988 7635.05 748.855 7632.36 742.891 7631.02 736.094 7631.02 724.875 7631.02 716.355 7634.45 710.539 7641.32 704.746 7648.21 701.848 7658.27 701.848 7671.5M742.402 7734.27V7734.27"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M879.898 7683.78C879.898 7695.34 877.527 7704.3 872.785 7710.66 868.07 7717.01 861.434 7720.19 852.879 7720.19 844.371 7720.19 837.746 7717.01 833.004 7710.66 828.285 7704.3 825.926 7695.34 825.926 7683.78 825.926 7672.27 828.285 7663.33 833.004 7656.98 837.746 7650.62 844.371 7647.45 852.879 7647.45 861.434 7647.45 868.07 7650.62 872.785 7656.98 877.527 7663.33 879.898 7672.27 879.898 7683.78M896.328 7646.09C896.328 7629.12 892.648 7616.51 885.289 7608.25 877.957 7599.96 866.727 7595.82 851.594 7595.82 845.973 7595.82 840.68 7596.21 835.719 7596.99 830.754 7597.75 825.926 7598.92 821.234 7600.51V7616.94C825.879 7614.54 830.461 7612.77 834.984 7611.62 839.531 7610.47 844.152 7609.9 848.844 7609.9 859.234 7609.9 867.004 7612.65 872.164 7618.15 877.32 7623.65 879.898 7631.95 879.898 7643.04V7649.79C876.625 7644.29 872.434 7640.17 867.324 7637.43 862.215 7634.72 856.09 7633.36 848.953 7633.36 837.121 7633.36 827.59 7637.96 820.355 7647.15 813.117 7656.37 809.5 7668.57 809.5 7683.78 809.5 7699.04 813.117 7711.25 820.355 7720.45 827.59 7729.66 837.121 7734.27 848.953 7734.27 856.09 7734.27 862.215 7732.9 867.324 7730.16 872.434 7727.45 876.625 7723.34 879.898 7717.84V7731.92H896.328V7646.09"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M989.289
7762.43H1008.06V7710.8H1069.07V7762.43H1087.85V7633.36H1069.07V7696.72H1008.06V7633.
36H989.289V7762.43"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1163.51 7720.19C1154.81 7720.19 1147.93 7716.84 1142.87 7710.14 1137.83 7703.45 1135.32 7694.28 1135.32 7682.64 1135.32 7671.01 1137.82 7661.84 1142.83 7655.14 1147.87 7648.45 1154.76 7645.1 1163.51 7645.1 1172.16 7645.1 1179.01 7648.46 1184.05 7655.18 1189.11 7661.9 1191.64 7671.06 1191.64 7682.64 1191.64 7694.18 1189.11 7703.32 1184.05 7710.07 1179.01 7716.82 1172.16 7720.19 1163.51 7720.19M1163.48 7734.27C1177.36 7734.27 1188.26 7729.7 1196.18 7720.55 1204.1 7711.44 1208.06 7698.8 1208.06 7682.64 1208.06 7666.54 1204.1 7653.9 1196.18 7644.73 1188.26 7635.59 1177.36 7631.02 1163.48 7631.02 1149.54 7631.02 1138.63 7635.59 1130.73 7644.73 1122.84 7653.9 1118.89 7666.54 1118.89 7682.64 1118.89 7698.8 1122.84 7711.44 1130.73 7720.55 1138.63 7729.7 1149.54 7734.27 1163.48 7734.27"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1298.09 7727.23V7713.15C1293.7 7715.5 1289.13 7717.26 1284.38 7718.43 1279.64 7719.6 1274.73 7720.19 1269.64 7720.19 1261.92 7720.19 1256.13 7718.98 1252.26 7716.56 1248.4 7714.14 1246.47 7710.51 1246.47 7705.67 1246.47 7701.98 1247.81 7699.08 1250.5 7696.98 1253.21 7694.88 1258.67 7692.88 1266.86 7690.96L1272.1 7689.76C1283.2 7687.31 1291.08 7683.86 1295.75 7679.38 1300.44 7674.91 1302.79 7668.66 1302.79 7660.64 1302.79 7651.53 1299.29 7644.3 1292.3 7638.97 1285.31 7633.67 1275.69 7631.02 1263.45 7631.02 1258.34 7631.02 1253.02 7631.61 1247.5 7632.78 1241.97 7633.95 1236.15 7635.71 1230.04 7638.06V7654.48C1235.79 7651.36 1241.45 7649.01 1247.02 7647.45 1252.59 7645.88 1258.11 7645.1 1263.55 7645.1 1270.89 7645.1 1276.52 7646.38 1280.46 7648.95 1284.39 7651.54 1286.36 7655.17 1286.36 7659.84 1286.36 7664.19 1284.97 7667.51 1282.18 7669.81 1279.39 7672.13 1273.27 7674.37 1263.81 7676.52L1258.5 7677.8C1248.45 7679.91 1241.19 7683.13 1236.71 7687.48 1232.27 7691.84 1230.04 7697.8 1230.04 7705.38 1230.04 7714.59 1233.27 7721.71 1239.72 7726.71 1246.18 7731.75 1255.33 7734.27 1267.18 7734.27 1273.05 7734.27 1278.58 7733.68 1283.76 7732.51 1288.94 7731.34 1293.72 7729.58 1298.09 7727.23"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1347.67
7760.08V7731.92H1380.52V7720.19H1347.67V7666.44C1347.67 7658.37 1348.75 7653.19 1350.9 7650.89 1353.07 7648.59 1357.48 7647.45 1364.13 7647.45H1380.52V7633.36H1364.13C1351.69 7633.36 1343.1 7635.73 1338.36 7640.48 1333.61 7645.24 1331.24 7653.9 1331.24 7666.44V7720.19H1319.51V7731.92H1331.24V7760.08H1347.67"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M318.688 6722.82H1714.848V7329.8469H318.688Z" fill="#ffffff" fill-rule="evenodd"/> <path transform="matrix(.1,0,0,-.1,0,796)" stroke-width="2.8346" stroke-linecap="butt" stroke-miterlimit="10" stroke-linejoin="miter" fill="none" stroke="#ffffff" d="M318.688 6722.82H1714.848V7329.8469H318.688Z"/> <path transform="matrix(.1,0,0,-.1,0,796)" stroke-

width="30.3515" stroke-linecap="butt" stroke-miterlimit="10" stroke-linejoin="miter" fill="none" stroke="#000000" d="M318.688 6722.82H1714.848V7329.8469H318.688Z"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M602.332 7185.71V7084.81H623.453C641.297 7084.81 654.359 7088.86 662.648 7096.98 670.934 7105.09 675.078 7117.9 675.078 7135.41 675.078 7152.79 670.934 7165.51 662.648 7173.57 654.359 7181.67 641.297 7185.71 623.453 7185.71H602.332M583.559 7199.79H620.773C645.953 7199.79 664.418 7194.59 676.176 7184.17 687.961 7173.79 693.852 7157.53 693.852 7135.41 693.852 7113.14 687.934 7096.79 676.105 7086.35 664.273 7075.93 645.828 7070.73 620.773 7070.73H583.559V7199.79"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M807.508 7124.99V7117.66H732.418C733.125 7106.2 736.477 7097.46 742.465 7091.44 748.453 7085.45 756.789 7082.46 767.469 7082.46 773.68 7082.46 779.691 7083.24 785.508 7084.81 791.328 7086.37 797.098 7088.72 802.816 7091.85V7077.77C797.047 7074.71 791.133 7072.38 785.07 7070.76 779.031 7069.18 772.91 7068.38 766.699 7068.38 751.105 7068.38 738.746 7072.94 729.629 7082.06 720.535 7091.2 715.992 7103.55 715.992 7119.13 715.992 7135.23 720.355 7148.01 729.082 7157.44 737.809 7166.9 749.578 7171.63 764.391 7171.63 777.688 7171.63 788.199 7167.45 795.922 7159.09 803.648 7150.73 807.508 7139.37 807.508 7124.99M791.082 7129.39C790.961 7137.95 788.457 7144.78 783.566 7149.89 778.703 7155 772.262 7157.55 764.242 7157.55 755.148 7157.55 747.867 7155.08 742.391 7150.14 736.938 7145.21 733.797 7138.25 732.969 7129.28L791.082 7129.39"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M905.289 7120.01C905.289 7131.71 902.883 7140.89 898.066 7147.54 893.277 7154.21 886.688 7157.55 878.305 7157.55 869.918 7157.55 863.32 7154.21 858.504 7147.54 853.715 7140.89 851.316 7131.71 851.316 7120.01 851.316 7108.3 853.715 7099.11 858.504 7092.43 863.32 7085.79 869.918 7082.46 878.305 7082.46 886.688 7082.46 893.277 7085.79 898.066 7092.43 902.883 7099.11 905.289 7108.3 905.289 7120.01M851.316 7155.21C854.641 7160.75 858.848 7164.88 863.93 7167.56 869.016 7170.28 875.09 7171.63 882.152 7171.63 893.887 7171.63 903.406 7166.89 910.719 7157.41 918.051 7147.92 921.719 7135.45 921.719 7120.01 921.719 7104.56 918.051 7092.09 910.719 7082.61 903.406 7073.12 893.887 7068.38 882.152 7068.38 875.09 7068.38 869.016 7069.72 863.93 7072.41 858.848 7075.13 854.641 7079.26 851.316 7084.81V7070.73H834.891V7206.83H851.316V7155.21"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M950.293 7108.86V7169.29H966.719V7109.48C966.719 7100.49 968.504 7093.73 972.074 7089.21 975.641 7084.71 981.008 7082.46 988.168 7082.46 996.75 7082.46 1003.53 7085.16 1008.52 7090.56 1013.5 7095.99 1016 7103.37 1016 7112.71V7169.29H1032.43V7070.73H1016V7084.81C1012.11 7079.26 1007.59 7075.13 1002.43 7072.41 997.301 7069.72 991.336 7068.38 984.539 7068.38 973.32 7068.38 964.801 7071.82 958.984 7078.68 953.191 7085.58 950.293 7095.64 950.293 7108.86M990.848 7171.63V7171.63"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1128.34 7121.14C1128.34 7132.7 1125.97 7141.66 1121.23 7148.02 1116.51 7154.38 1109.88 7157.55 1101.32 7157.55 1092.82 7157.55 1086.19 7154.38 1081.45 7148.02 1076.73 7141.66 1074.37 7132.71 1074.37 7121.14 1074.37 7109.63 1076.73 7100.7 1081.45 7094.34 1086.19 7087.98 1092.82 7084.81 1101.32 7084.81 1109.88 7084.81 1116.51 7087.98 1121.23 7094.34 1125.97 7100.7 1128.34 7109.63 1128.34 7121.14M1144.77 7083.45C1144.77 7066.48 1141.09 7053.87 1133.73 7045.61 1126.4 7037.32 1115.17 7033.18 1100.04 7033.18 1094.41 7033.18 1089.13 7033.57 1084.16 7034.36 1079.2 7035.11 1074.37 7036.29 1069.68 7037.88V7054.3C1074.32 7051.91 1078.91 7050.13 1083.43 7048.98 1087.97 7047.84 1092.59 7047.26 1097.29 7047.26 1107.68 7047.26 1115.45 7050.01 1120.61 7055.51 1125.77 7061.01 1128.34 7069.31 1128.34 7080.41V7087.15C1125.07 7081.65 1120.88 7077.54 1115.77 7074.8 1110.66 7072.08 1104.54 7070.73 1097.4 7070.73 1085.57 7070.73 1076.04 7075.32 1068.8 7084.51 1061.56 7093.73 1057.95 7105.94 1057.95 7121.14 1057.95 7136.39 1061.56 7148.62 1068.8 7157.81 1076.04 7167.02 1085.57 7171.63 1097.4 7171.63 1104.54 7171.63 1110.66 7170.26 1115.77 7167.53 1120.88 7164.81 1125.07 7160.71 1128.34 7155.21V7169.29H1144.77V7083.45"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1243.75 7121.14C1243.75 7132.7 1241.38 7141.66 1236.63 7148.02 1231.91 7154.38 1225.28 7157.55 1216.72 7157.55 1208.22 7157.55 1201.59 7154.38 1196.85 7148.02 1192.13 7141.66 1189.77 7132.71 1189.77 7121.14 1189.77 7109.63 1192.13 7100.7 1196.85 7094.34 1201.59 7087.98 1208.22 7084.81 1216.72 7084.81 1225.28 7084.81 1231.91 7087.98 1236.63 7094.34 1241.38 7100.7 1243.75 7109.63 1243.75 7121.14M1260.17 7083.45C1260.17 7066.48 1256.5 7053.87


1249.14 7045.61 1241.8 7037.32 1230.57 7033.18 1215.44 7033.18 1209.82 7033.18 1204.53 7033.57 1199.56 7034.36 1194.6 7035.11 1189.77 7036.29 1185.08 7037.88V7054.3C1189.73 7051.91 1194.31 7050.13 1198.83 7048.98 1203.38 7047.84 1208 7047.26 1212.69 7047.26 1223.08 7047.26 1230.85 7050.01 1236.01 7055.51 1241.17 7061.01 1243.75 7069.31 1243.75 7080.41V7087.15C1240.47 7081.65 1236.28 7077.54 1231.17 7074.8 1226.06 7072.08 1219.94 7070.73 1212.8 7070.73 1200.97 7070.73 1191.44 7075.32 1184.2 7084.51 1176.96 7093.73 1173.35 7105.94 1173.35 7121.14 1173.35 7136.39 1176.96 7148.62 1184.2 7157.81 1191.44 7167.02 1200.97 7171.63 1212.8 7171.63 1219.94 7171.63 1226.06 7170.26 1231.17 7167.53 1236.28 7164.81 1240.47 7160.71 1243.75 7155.21V7169.29H1260.17V7083.45"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1380.27 7124.99V7117.66H1305.18C1305.88 7106.2 1309.23 7097.46 1315.22 7091.44 1321.21 7085.45 1329.55 7082.46 1340.23 7082.46 1346.44 7082.46 1352.45 7083.24 1358.27 7084.81 1364.09 7086.37 1369.86 7088.72 1375.57 7091.85V7077.77C1369.8 7074.71 1363.89 7072.38 1357.83 7070.76 1351.79 7069.18 1345.67 7068.38 1339.46 7068.38 1323.86 7068.38 1311.51 7072.94 1302.39 7082.06 1293.3 7091.2 1288.75 7103.55 1288.75 7119.13 1288.75 7135.23 1293.11 7148.01 1301.84 7157.44 1310.57 7166.9 1322.34 7171.63 1337.15 7171.63 1350.45 7171.63 1360.96 7167.45 1368.68 7159.09 1376.41 7150.73 1380.27 7139.37 1380.27 7124.99M1363.84 7129.39C1363.72 7137.95 1361.21 7144.78 1356.32 7149.89 1351.46 7155 1345.02 7157.55 1337 7157.55 1327.91 7157.55 1320.63 7155.08 1315.15 7150.14 1309.7 7145.21 1306.56 7138.25 1305.73 7129.28L1363.84 7129.39"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1463.97 7155.21C1462.21 7156.01 1460.3 7156.6 1458.25 7156.96 1456.2 7157.36 1453.92 7157.55 1451.43 7157.55 1442.63 7157.55 1435.87 7154.61 1431.15 7148.71 1426.43 7142.85 1424.08 7134.42 1424.08 7123.42V7070.73H1407.65V7169.29H1424.08V7155.21C1427.35 7160.75 1431.62 7164.88 1436.87 7167.56 1442.13 7170.28 1448.51 7171.63 1456.01 7171.63 1457.09 7171.63 1458.27 7171.63 1459.57 7171.63 1460.86 7171.63 1462.31 7171.63 1463.89 7171.63L1463.97 7155.21"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M657.82 6980.06C649.875 6966.64 643.973 6953.37 640.109 6940.25 636.273 6927.14 634.355 6913.84 634.355 6900.35 634.355 6886.88 636.297 6873.54 640.184 6860.31 644.07 6847.11 649.949 6833.84 657.82 6820.49H643.594C635.016 6834.21 628.586 6847.68 624.309 6860.9 620.055 6874.15 617.93 6887.3 617.93 6900.35 617.93 6913.36 620.043 6926.45 624.27 6939.62 628.5 6952.82 634.941 6966.3 643.594 6980.06H657.82"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M773.207 6898.23V6890.89H698.113C698.824 6879.43 702.172 6870.69 708.16 6864.68 714.148 6858.69 722.484 6855.69 733.168 6855.69 739.375 6855.69 745.387 6856.47 751.207 6858.04 757.023 6859.6 762.793 6861.95 768.512 6865.08V6851C762.742 6847.94 756.828 6845.61 750.766 6844 744.727 6842.41 738.605 6841.61 732.395 6841.61 716.801 6841.61 704.445 6846.17 695.328 6855.29 686.234 6864.43 681.688 6876.79 681.688 6892.36 681.688 6908.47 686.051 6921.24 694.777 6930.68 703.504 6940.13 715.273 6944.86 730.086 6944.86 743.383 6944.86 753.895 6940.68 761.621 6932.32 769.344 6923.96 773.207 6912.6 773.207 6898.23M756.781 6902.63C756.656 6911.18 754.152 6918.01 749.262 6923.12 744.398 6928.23 737.957 6930.79 729.941 6930.79 720.848 6930.79 713.563 6928.32 708.086 6923.38 702.637 6918.44 699.496 6911.48 698.664 6902.52L756.781 6902.63"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M863.945 6894.38C863.945 6905.94 861.574 6914.89 856.832 6921.25 852.117 6927.61 845.48 6930.79 836.922 6930.79 828.418 6930.79 821.793 6927.61 817.051 6921.25 812.332 6914.89 809.973 6905.94 809.973 6894.38 809.973 6882.86 812.332 6873.93 817.051 6867.57 821.793 6861.21 828.418 6858.04 836.922 6858.04 845.48 6858.04 852.117 6861.21 856.832 6867.57 861.574 6873.93 863.945 6882.86 863.945 6894.38M880.375 6856.68C880.375 6839.72 876.695 6827.11 869.336 6818.84 862.004 6810.55 850.77 6806.41 835.641 6806.41 830.02 6806.41 824.727 6806.8 819.766 6807.59 814.801 6808.34 809.973 6809.52 805.281 6811.11V6827.53C809.926 6825.14 814.508 6823.36 819.031 6822.21 823.578 6821.07 828.195 6820.49 832.891 6820.49 843.277 6820.49 851.051 6823.24 856.211 6828.74 861.367 6834.24 863.945 6842.54 863.945 6853.64V6860.39C860.672 6854.89 856.48 6850.77 851.371 6848.03 846.262 6845.32 840.137 6843.96 833 6843.96 821.168 6843.96 811.637 6848.55 804.402 6857.75 797.164 6866.96 793.547 6879.17 793.547 6894.38 793.547 6909.63 797.164 6921.85 804.402 6931.04 811.637 6940.26 821.168 6944.86 833 6944.86 840.137 6944.86 846.262 6943.5 851.371 6940.76 856.48 6938.04 860.672 6933.94 863.945

6928.44V6942.52H880.375V6856.68"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M920.684 6865.08H939.457V6843.96H920.684V6865.08"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1094.04 6894.38C1094.04 6905.94 1091.67 6914.89 1086.93 6921.25 1082.21 6927.61 1075.57 6930.79 1067.02 6930.79 1058.51 6930.79 1051.89 6927.61 1047.14 6921.25 1042.43 6914.89 1040.07 6905.94 1040.07 6894.38 1040.07 6882.86 1042.43 6873.93 1047.14 6867.57 1051.89 6861.21 1058.51 6858.04 1067.02 6858.04 1075.57 6858.04 1082.21 6861.21 1086.93 6867.57 1091.67 6873.93 1094.04 6882.86 1094.04 6894.38M1110.47 6856.68C1110.47 6839.72 1106.79 6827.11 1099.43 6818.84 1092.1 6810.55 1080.87 6806.41 1065.73 6806.41 1060.11 6806.41 1054.82 6806.8 1049.86 6807.59 1044.89 6808.34 1040.07 6809.52 1035.38 6811.11V6827.53C1040.02 6825.14 1044.6 6823.36 1049.13 6822.21 1053.67 6821.07 1058.29 6820.49 1062.98 6820.49 1073.38 6820.49 1081.14 6823.24 1086.3 6828.74 1091.46 6834.24 1094.04 6842.54 1094.04 6853.64V6860.39C1090.77 6854.89 1086.57 6850.77 1081.46 6848.03 1076.36 6845.32 1070.23 6843.96 1063.09 6843.96 1051.26 6843.96 1041.73 6848.55 1034.5 6857.75 1027.26 6866.96 1023.64 6879.17 1023.64 6894.38 1023.64 6909.63 1027.26 6921.85 1034.5 6931.04 1041.73 6940.26 1051.26 6944.86 1063.09 6944.86 1070.23 6944.86 1076.36 6943.5 1081.46 6940.76 1086.57 6938.04 1090.77 6933.94 1094.04 6928.44V6942.52H1110.47V6856.68"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1209.44 6928.44V6980.06H1225.87V6843.96H1209.44V6858.04C1206.12 6852.49 1201.91 6848.36 1196.83 6845.64 1191.75 6842.96 1185.63 6841.61 1178.5 6841.61 1166.84 6841.61 1157.34 6846.36 1150.01 6855.84 1142.7 6865.32 1139.04 6877.79 1139.04 6893.24 1139.04 6908.69 1142.7 6921.15 1150.01 6930.64 1157.34 6940.12 1166.84 6944.86 1178.5 6944.86 1185.63 6944.86 1191.75 6943.51 1196.83 6940.79 1201.91 6938.11 1206.12 6933.99 1209.44 6928.44M1155.47 6893.24C1155.47 6881.53 1157.87 6872.34 1162.66 6865.66 1167.45 6859.02 1174.04 6855.69 1182.42 6855.69 1190.8 6855.69 1197.4 6859.02 1202.22 6865.66 1207.04 6872.34 1209.44 6881.53 1209.44 6893.24 1209.44 6904.95 1207.04 6914.13 1202.22 6920.77 1197.4 6927.45 1190.8 6930.79 1182.42 6930.79 1174.04 6930.79 1167.45 6927.45 1162.66 6920.77 1157.87 6914.13 1155.47 6904.95 1155.47 6893.24"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1331.89 6893.24C1331.89 6904.95 1329.48 6914.13 1324.66 6920.77 1319.87 6927.45 1313.28 6930.79 1304.9 6930.79 1296.52 6930.79 1289.91 6927.45 1285.1 6920.77 1280.31 6914.13 1277.91 6904.95 1277.91 6893.24 1277.91 6881.53 1280.31 6872.34 1285.1 6865.66 1289.91 6859.02 1296.52 6855.69 1304.9 6855.69 1313.28 6855.69 1319.87 6859.02 1324.66 6865.66 1329.48 6872.34 1331.89 6881.53 1331.89 6893.24M1277.91 6928.44C1281.24 6933.99 1285.44 6938.11 1290.52 6940.79 1295.61 6943.51 1301.68 6944.86 1308.75 6944.86 1320.48 6944.86 1330 6940.12 1337.31 6930.64 1344.64 6921.15 1348.31 6908.69 1348.31 6893.24 1348.31 6877.79 1344.64 6865.32 1337.31 6855.84 1330 6846.36 1320.48 6841.61 1308.75 6841.61 1301.68 6841.61 1295.61 6842.96 1290.52 6845.64 1285.44 6848.36 1281.24 6852.49 1277.91 6858.04V6843.96H1261.48V6980.06H1277.91V6928.44"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1374.54 6980.06H1388.77C1397.35 6966.3 1403.77 6952.82 1408.02 6939.62 1412.3 6926.45 1414.43 6913.36 1414.43 6900.35 1414.43 6887.3 1412.3 6874.15 1408.02 6860.9 1403.77 6847.68 1397.35 6834.21 1388.77 6820.49H1374.54C1382.41 6833.84 1388.29 6847.11 1392.18 6860.31 1396.06 6873.54 1398.01 6886.88 1398.01 6900.35 1398.01 6913.84 1396.06 6927.14 1392.18 6940.25 1388.29 6953.37 1382.41 6966.64 1374.54 6980.06"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M2443.28 6722.82H4567.88V7329.8469H2443.28Z" fill="#ffffff" fill-rule="evenodd"/> <path transform="matrix(.1,0,0,-.1,0,796)" stroke-width="2.8346" stroke-linecap="butt" stroke-miterlimit="10" stroke-linejoin="miter" fill="none" stroke="#ffffff" d="M2443.28 6722.82H4567.88V7329.8469H2443.28Z"/> <path transform="matrix(.1,0,0,-.1,0,796)" stroke-width="30.3515" stroke-linecap="butt" stroke-miterlimit="10" stroke-linejoin="miter" fill="none" stroke="#000000" d="M2443.28 6722.82H4567.88V7329.8469H2443.28Z"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M2765.16 7185.71V7084.81H2786.28C2804.12 7084.81 2817.19 7088.86 2825.47 7096.98 2833.76 7105.09 2837.9 7117.9 2837.9 7135.41 2837.9 7152.79 2833.76 7165.51 2825.47 7173.57 2817.19 7181.67 2804.12 7185.71 2786.28 7185.71H2765.16M2746.38 7199.79H2783.6C2808.78 7199.79 2827.25 7194.59 2839 7184.17 2850.79 7173.79 2856.68 7157.53 2856.68 7135.41 2856.68 7113.14 2850.76 7096.79 2838.93 7086.35 2827.1 7075.93 2808.66 7070.73

2783.6 7070.73H2746.38V7199.79"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M2970.34 7124.99V7117.66H2895.24C2895.95 7106.2 2899.3 7097.46 2905.29 7091.44 2911.28 7085.45 2919.61 7082.46 2930.3 7082.46 2936.5 7082.46 2942.52 7083.24 2948.34 7084.81 2954.15 7086.37 2959.92 7088.72 2965.64 7091.85V7077.77C2959.88 7074.71 2953.96 7072.38 2947.89 7070.76 2941.86 7069.18 2935.73 7068.38 2929.53 7068.38 2913.93 7068.38 2901.57 7072.94 2892.46 7082.06 2883.36 7091.2 2878.82 7103.55 2878.82 7119.13 2878.82 7135.23 2883.18 7148.01 2891.91 7157.44 2900.63 7166.9 2912.4 7171.63 2927.21 7171.63 2940.51 7171.63 2951.02 7167.45 2958.75 7159.09 2966.47 7150.73 2970.34 7139.37 2970.34 7124.99M2953.91 7129.39C2953.79 7137.95 2951.28 7144.78 2946.39 7149.89 2941.53 7155 2935.09 7157.55 2927.07 7157.55 2917.98 7157.55 2910.69 7155.08 2905.21 7150.14 2899.77 7145.21 2896.63 7138.25 2895.79 7129.28L2953.91 7129.39"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M3068.12 7120.01C3068.12 7131.71 3065.71 7140.89 3060.89 7147.54 3056.1 7154.21 3049.52 7157.55 3041.13 7157.55 3032.75 7157.55 3026.14 7154.21 3021.33 7147.54 3016.54 7140.89 3014.14 7131.71 3014.14 7120.01 3014.14 7108.3 3016.54 7099.11 3021.33 7092.43 3026.14 7085.79 3032.75 7082.46 3041.13 7082.46 3049.52 7082.46 3056.1 7085.79 3060.89 7092.43 3065.71 7099.11 3068.12 7108.3 3068.12 7120.01M3014.14 7155.21C3017.47 7160.75 3021.67 7164.88 3026.76 7167.56 3031.84 7170.28 3037.91 7171.63 3044.98 7171.63 3056.71 7171.63 3066.23 7166.89 3073.54 7157.41 3080.88 7147.92 3084.54 7135.45 3084.54 7120.01 3084.54 7104.56 3080.88 7092.09 3073.54 7082.61 3066.23 7073.12 3056.71 7068.38 3044.98 7068.38 3037.91 7068.38 3031.84 7069.72 3026.76 7072.41 3021.67 7075.13 3017.47 7079.26 3014.14 7084.81V7070.73H2997.71V7206.83H3014.14V7155.21"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M3113.12 7108.86V7169.29H3129.55V7109.48C3129.55 7100.49 3131.33 7093.73 3134.9 7089.21 3138.47 7084.71 3143.83 7082.46 3151 7082.46 3159.57 7082.46 3166.36 7085.16 3171.34 7090.56 3176.33 7095.99 3178.82 7103.37 3178.82 7112.71V7169.29H3195.25V7070.73H3178.82V7084.81C3174.94 7079.26 3170.41 7075.13 3165.26 7072.41 3160.13 7069.72 3154.16 7068.38 3147.36 7068.38 3136.14 7068.38 3127.63 7071.82 3121.81 7078.68 3116.02 7085.58 3113.12 7095.64 3113.12 7108.86M3153.67 7171.63V7171.63"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M3291.17 7121.14C3291.17 7132.7 3288.8 7141.66 3284.06 7148.02 3279.34 7154.38 3272.7 7157.55 3264.15 7157.55 3255.64 7157.55 3249.02 7154.38 3244.27 7148.02 3239.55 7141.66 3237.2 7132.71 3237.2 7121.14 3237.2 7109.63 3239.55 7100.7 3244.27 7094.34 3249.02 7087.98 3255.64 7084.81 3264.15 7084.81 3272.7 7084.81 3279.34 7087.98 3284.06 7094.34 3288.8 7100.7 3291.17 7109.63 3291.17 7121.14M3307.6 7083.45C3307.6 7066.48 3303.92 7053.87 3296.56 7045.61 3289.23 7037.32 3278 7033.18 3262.86 7033.18 3257.24 7033.18 3251.95 7033.57 3246.99 7034.36 3242.02 7035.11 3237.2 7036.29 3232.5 7037.88V7054.3C3237.15 7051.91 3241.73 7050.13 3246.25 7048.98 3250.8 7047.84 3255.42 7047.26 3260.11 7047.26 3270.5 7047.26 3278.28 7050.01 3283.43 7055.51 3288.59 7061.01 3291.17 7069.31 3291.17 7080.41V7087.15C3287.89 7081.65 3283.7 7077.54 3278.59 7074.8 3273.48 7072.08 3267.36 7070.73 3260.22 7070.73 3248.39 7070.73 3238.86 7075.32 3231.63 7084.51 3224.39 7093.73 3220.77 7105.94 3220.77 7121.14 3220.77 7136.39 3224.39 7148.62 3231.63 7157.81 3238.86 7167.02 3248.39 7171.63 3260.22 7171.63 3267.36 7171.63 3273.48 7170.26 3278.59 7167.53 3283.7 7164.81 3287.89 7160.71 3291.17 7155.21V7169.29H3307.6V7083.45"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M3381.79 7199.79H3494.43V7185.71H3447.49V7070.73H3428.72V7185.71H3381.79V7199.79"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M3540.42 7155.21C3538.66 7156.01 3536.75 7156.6 3534.7 7156.96 3532.65 7157.36 3530.38 7157.55 3527.88 7157.55 3519.08 7157.55 3512.32 7154.61 3507.61 7148.71 3502.89 7142.85 3500.53 7134.42 3500.53 7123.42V7070.73H3484.1V7169.29H3500.53V7155.21C3503.8 7160.75 3508.07 7164.88 3513.32 7167.56 3518.58 7170.28 3524.96 7171.63 3532.46 7171.63 3533.54 7171.63 3534.72 7171.63 3536.02 7171.63 3537.32 7171.63 3538.76 7171.63 3540.35 7171.63L3540.42 7155.21"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M3601.48 7120.01C3588.82 7120.01 3580.05 7118.55 3575.16 7115.64 3570.27 7112.73 3567.82 7107.76 3567.82 7100.72 3567.82 7095.12 3569.66 7090.67 3573.32 7087.38 3576.99 7084.1 3581.98 7082.46 3588.29 7082.46 3596.99 7082.46 3603.96 7085.54 3609.22 7091.7 3614.48 7097.88 3617.11 7106.1 3617.11 7116.34V7120.01H3601.48M3633.53


7126.2V7070.73H3617.11V7084.81C3613.49 7079.18 3608.98 7075.04 3603.57 7072.38 3598.17 7069.71 3591.56 7068.38 3583.74 7068.38 3573.84 7068.38 3565.97 7071.14 3560.13 7076.67 3554.31 7082.21 3551.4 7089.63 3551.4 7098.92 3551.4 7109.78 3555.05 7117.95 3562.36 7123.45 3569.67 7128.98 3580.59 7131.74 3595.11 7131.74H3617.11V7133.5C3617.11 7141.15 3614.7 7147.07 3609.88 7151.25 3605.09 7155.45 3598.36 7157.55 3589.68 7157.55 3584.15 7157.55 3578.78 7156.96 3573.54 7155.79 3568.31 7154.62 3563.28 7152.86 3558.44 7150.51V7164.59C3564.28 7166.94 3569.95 7168.7 3575.45 7169.87 3580.98 7171.05 3586.34 7171.63 3591.55 7171.63 3605.63 7171.63 3616.14 7167.87 3623.08 7160.34 3630.05 7152.84 3633.53 7141.46 3633.53 7126.2"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M3751.72 7131.08V7070.73H3735.3V7130.57C3735.3 7139.59 3733.5 7146.33 3729.91 7150.8 3726.34 7155.3 3720.97 7157.55 3713.81 7157.55 3705.21 7157.55 3698.42 7154.84 3693.46 7149.41 3688.5 7144.01 3686.02 7136.63 3686.02 7127.27V7070.73H3669.59V7169.29H3686.02V7155.21C3689.86 7160.71 3694.38 7164.81 3699.59 7167.53 3704.79 7170.26 3710.79 7171.63 3717.59 7171.63 3728.83 7171.63 3737.32 7168.2 3743.07 7161.33 3748.84 7154.48 3751.72 7144.4 3751.72 7131.08"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M3845.3 7164.59V7150.51C3840.89 7152.86 3836.32 7154.62 3831.58 7155.79 3826.84 7156.96 3821.93 7157.55 3816.84 7157.55 3809.12 7157.55 3803.32 7156.34 3799.46 7153.92 3795.6 7151.5 3793.67 7147.87 3793.67 7143.03 3793.67 7139.34 3795.02 7136.45 3797.7 7134.34 3800.42 7132.24 3805.87 7130.23 3814.05 7128.33L3819.3 7127.12C3830.4 7124.68 3838.28 7121.21 3842.95 7116.74 3847.64 7112.27 3849.99 7106.02 3849.99 7098.01 3849.99 7088.89 3846.49 7081.66 3839.5 7076.34 3832.51 7071.03 3822.89 7068.38 3810.64 7068.38 3805.54 7068.38 3800.22 7068.97 3794.7 7070.14 3789.17 7071.31 3783.36 7073.07 3777.24 7075.42V7091.85C3782.99 7088.72 3788.64 7086.37 3794.22 7084.81 3799.79 7083.24 3805.3 7082.46 3810.76 7082.46 3818.09 7082.46 3823.72 7083.74 3827.66 7086.31 3831.59 7088.9 3833.56 7092.53 3833.56 7097.2 3833.56 7101.55 3832.17 7104.88 3829.38 7107.17 3826.6 7109.5 3820.47 7111.73 3811.01 7113.88L3805.7 7115.17C3795.65 7117.27 3788.39 7120.5 3783.92 7124.85 3779.47 7129.2 3777.24 7135.16 3777.24 7142.74 3777.24 7151.95 3780.47 7159.07 3786.92 7164.08 3793.38 7169.11 3802.53 7171.63 3814.39 7171.63 3820.25 7171.63 3825.78 7171.05 3830.96 7169.87 3836.14 7168.7 3840.92 7166.94 3845.3 7164.59"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M3878.45 7206.83H3894.87V7070.73H3878.45V7206.83"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M3971.75 7120.01C3959.09 7120.01 3950.32 7118.55 3945.43 7115.64 3940.54 7112.73 3938.1 7107.76 3938.1 7100.72 3938.1 7095.12 3939.93 7090.67 3943.6 7087.38 3947.26 7084.1 3952.25 7082.46 3958.55 7082.46 3967.26 7082.46 3974.24 7085.54 3979.49 7091.7 3984.75 7097.88 3987.38 7106.1 3987.38 7116.34V7120.01H3971.75M4003.8 7126.2V7070.73H3987.38V7084.81C3983.76 7079.18 3979.25 7075.04 3973.85 7072.38 3968.45 7069.71 3961.83 7068.38 3954.01 7068.38 3944.11 7068.38 3936.24 7071.14 3930.39 7076.67 3924.58 7082.21 3921.67 7089.63 3921.67 7098.92 3921.67 7109.78 3925.32 7117.95 3932.63 7123.45 3939.94 7128.98 3950.86 7131.74 3965.38 7131.74H3987.38V7133.5C3987.38 7141.15 3984.97 7147.07 3980.15 7151.25 3975.36 7155.45 3968.63 7157.55 3959.95 7157.55 3954.43 7157.55 3949.05 7156.96 3943.82 7155.79 3938.59 7154.62 3933.55 7152.86 3928.71 7150.51V7164.59C3934.55 7166.94 3940.22 7168.7 3945.72 7169.87 3951.25 7171.05 3956.61 7171.63 3961.82 7171.63 3975.9 7171.63 3986.41 7167.87 3993.35 7160.34 4000.32 7152.84 4003.8 7141.46 4003.8 7126.2"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M4056.29 7197.45V7169.29H4089.14V7157.55H4056.29V7103.8C4056.29 7095.73 4057.36 7090.55 4059.52 7088.25 4061.69 7085.96 4066.1 7084.81 4072.75 7084.81H4089.14V7070.73H4072.75C4060.31 7070.73 4051.72 7073.1 4046.98 7077.84 4042.23 7082.61 4039.86 7091.26 4039.86 7103.8V7157.55H4028.13V7169.29H4039.86V7197.45H4056.29"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M4148.25 7157.55C4139.54 7157.55 4132.66 7154.2 4127.6 7147.51 4122.57 7140.81 4120.05 7131.64 4120.05 7120.01 4120.05 7108.37 4122.55 7099.2 4127.57 7092.51 4132.6 7085.81 4139.49 7082.46 4148.25 7082.46 4156.9 7082.46 4163.74 7085.82 4168.78 7092.54 4173.84 7099.27 4176.37 7108.42 4176.37 7120.01 4176.37 7131.54 4173.84 7140.69 4168.78 7147.43 4163.74 7154.18 4156.9 7157.55 4148.25 7157.55M4148.21 7171.63C4162.09 7171.63 4172.99 7167.06 4180.91 7157.92 4188.83 7148.8 4192.79 7136.16 4192.79 7120.01 4192.79 7103.9 4188.83 7091.26 4180.91 7082.09 4172.99 7072.95 4162.09 7068.38 4148.21 7068.38 4134.27 7068.38 4123.36

7072.95 4115.46 7082.09 4107.57 7091.26 4103.62 7103.9 4103.62 7120.01 4103.62 7136.16 4107.57 7148.8 4115.46 7157.92 4123.36 7167.06 4134.27 7171.63 4148.21 7171.63"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M4278.13 7155.21C4276.38 7156.01 4274.46 7156.6 4272.41 7156.96 4270.36 7157.36 4268.09 7157.55 4265.59 7157.55 4256.79 7157.55 4250.04 7154.61 4245.32 7148.71 4240.6 7142.85 4238.24 7134.42 4238.24 7123.42V7070.73H4221.81V7169.29H4238.24V7155.21C4241.52 7160.75 4245.78 7164.88 4251.04 7167.56 4256.29 7170.28 4262.67 7171.63 4270.18 7171.63 4271.25 7171.63 4272.44 7171.63 4273.73 7171.63 4275.03 7171.63 4276.47 7171.63 4278.06 7171.63L4278.13 7155.21"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M2872.38 6980.06C2864.43 6966.64 2858.53 6953.37 2854.67 6940.25 2850.83 6927.14 2848.91 6913.84 2848.91 6900.35 2848.91 6886.88 2850.86 6873.54 2854.74 6860.31 2858.63 6847.11 2864.51 6833.84 2872.38 6820.49H2858.15C2849.57 6834.21 2843.14 6847.68 2838.86 6860.9 2834.61 6874.15 2832.48 6887.3 2832.48 6900.35 2832.48 6913.36 2834.6 6926.45 2838.83 6939.62 2843.06 6952.82 2849.5 6966.3 2858.15 6980.06H2872.38"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M2987.76 6898.23V6890.89H2912.67C2913.38 6879.43 2916.73 6870.69 2922.72 6864.68 2928.71 6858.69 2937.04 6855.69 2947.72 6855.69 2953.93 6855.69 2959.95 6856.47 2965.76 6858.04 2971.58 6859.6 2977.35 6861.95 2983.07 6865.08V6851C2977.3 6847.94 2971.39 6845.61 2965.32 6844 2959.29 6842.41 2953.16 6841.61 2946.95 6841.61 2931.36 6841.61 2919 6846.17 2909.88 6855.29 2900.79 6864.43 2896.24 6876.79 2896.24 6892.36 2896.24 6908.47 2900.61 6921.24 2909.33 6930.68 2918.06 6940.13 2929.83 6944.86 2944.64 6944.86 2957.94 6944.86 2968.45 6940.68 2976.18 6932.32 2983.9 6923.96 2987.76 6912.6 2987.76 6898.23M2971.34 6902.63C2971.21 6911.18 2968.71 6918.01 2963.82 6923.12 2958.96 6928.23 2952.52 6930.79 2944.5 6930.79 2935.4 6930.79 2928.12 6928.32 2922.64 6923.38 2917.19 6918.44 2914.05 6911.48 2913.22 6902.52L2971.34 6902.63"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M3078.5 6894.38C3078.5 6905.94 3076.13 6914.89 3071.39 6921.25 3066.67 6927.61 3060.04 6930.79 3051.48 6930.79 3042.97 6930.79 3036.35 6927.61 3031.61 6921.25 3026.89 6914.89 3024.53 6905.94 3024.53 6894.38 3024.53 6882.86 3026.89 6873.93 3031.61 6867.57 3036.35 6861.21 3042.97 6858.04 3051.48 6858.04 3060.04 6858.04 3066.67 6861.21 3071.39 6867.57 3076.13 6873.93 3078.5 6882.86 3078.5 6894.38M3094.93 6856.68C3094.93 6839.72 3091.25 6827.11 3083.89 6818.84 3076.56 6810.55 3065.33 6806.41 3050.2 6806.41 3044.57 6806.41 3039.29 6806.8 3034.32 6807.59 3029.36 6808.34 3024.53 6809.52 3019.84 6811.11V6827.53C3024.48 6825.14 3029.07 6823.36 3033.59 6822.21 3038.13 6821.07 3042.75 6820.49 3047.45 6820.49 3057.84 6820.49 3065.61 6823.24 3070.77 6828.74 3075.93 6834.24 3078.5 6842.54 3078.5 6853.64V6860.39C3075.23 6854.89 3071.04 6850.77 3065.93 6848.03 3060.82 6845.32 3054.7 6843.96 3047.56 6843.96 3035.73 6843.96 3026.2 6848.55 3018.96 6857.75 3011.72 6866.96 3008.11 6879.17 3008.11 6894.38 3008.11 6909.63 3011.72 6921.85 3018.96 6931.04 3026.2 6940.26 3035.73 6944.86 3047.56 6944.86 3054.7 6944.86 3060.82 6943.5 3065.93 6940.76 3071.04 6938.04 3075.23 6933.94 3078.5 6928.44V6942.52H3094.93V6856.68"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M3135.24 6865.08H3154.01V6843.96H3135.24V6865.08"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M3298.11 6961.29C3285.52 6961.29 3275.51 6956.56 3268.08 6947.1 3260.68 6937.64 3256.97 6924.76 3256.97 6908.45 3256.97 6892.2 3260.68 6879.34 3268.08 6869.88 3275.51 6860.42 3285.52 6855.69 3298.11 6855.69 3310.72 6855.69 3320.71 6860.42 3328.07 6869.88 3335.43 6879.34 3339.11 6892.2 3339.11 6908.45 3339.11 6924.76 3335.43 6937.64 3328.07 6947.1 3320.71 6956.56 3310.72 6961.29 3298.11 6961.29M3298.11 6975.37C3316.25 6975.37 3330.75 6969.31 3341.6 6957.18 3352.45 6945.09 3357.88 6928.84 3357.88 6908.45 3357.88 6888.12 3352.45 6871.89 3341.6 6859.76 3330.75 6847.66 3316.25 6841.61 3298.11 6841.61 3279.93 6841.61 3265.39 6847.65 3254.52 6859.73 3243.64 6871.82 3238.2 6888.07 3238.2 6908.45 3238.2 6928.84 3243.64 6945.09 3254.52 6957.18 3265.39 6969.31 3279.93 6975.37 3298.11 6975.37"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M3404.68 6858.04V6806.41H3388.25V6942.52H3404.68V6928.44C3408 6933.99 3412.21 6938.11 3417.29 6940.79 3422.38 6943.51 3428.45 6944.86 3435.52 6944.86 3447.25 6944.86 3456.77 6940.12 3464.08 6930.64 3471.41 6921.15 3475.08 6908.69 3475.08 6893.24 3475.08 6877.79


3471.41 6865.32 3464.08 6855.84 3456.77 6846.36 3447.25 6841.61 3435.52 6841.61 3428.45 6841.61 3422.38 6842.96 3417.29 6845.64 3412.21 6848.36 3408 6852.49 3404.68 6858.04M3458.65 6893.24C3458.65 6904.95 3456.24 6914.13 3451.43 6920.77 3446.64 6927.45 3440.05 6930.79 3431.66 6930.79 3423.28 6930.79 3416.68 6927.45 3411.86 6920.77 3407.07 6914.13 3404.68 6904.95 3404.68 6893.24 3404.68 6881.53 3407.07 6872.34 3411.86 6865.66 3416.68 6859.02 3423.28 6855.69 3431.66 6855.69 3440.05 6855.69 3446.64 6859.02 3451.43 6865.66 3456.24 6872.34 3458.65 6881.53 3458.65 6893.24"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M3588.13 6898.23V6890.89H3513.04C3513.75 6879.43 3517.1 6870.69 3523.09 6864.68 3529.07 6858.69 3537.41 6855.69 3548.09 6855.69 3554.3 6855.69 3560.32 6856.47 3566.13 6858.04 3571.95 6859.6 3577.72 6861.95 3583.44 6865.08V6851C3577.67 6847.94 3571.75 6845.61 3565.7 6844 3559.66 6842.41 3553.53 6841.61 3547.32 6841.61 3531.73 6841.61 3519.37 6846.17 3510.25 6855.29 3501.16 6864.43 3496.61 6876.79 3496.61 6892.36 3496.61 6908.47 3500.98 6921.24 3509.7 6930.68 3518.43 6940.13 3530.2 6944.86 3545.02 6944.86 3558.31 6944.86 3568.82 6940.68 3576.55 6932.32 3584.27 6923.96 3588.13 6912.6 3588.13 6898.23M3571.71 6902.63C3571.59 6911.18 3569.08 6918.01 3564.19 6923.12 3559.32 6928.23 3552.89 6930.79 3544.87 6930.79 3535.77 6930.79 3528.49 6928.32 3523.02 6923.38 3517.56 6918.44 3514.42 6911.48 3513.59 6902.52L3571.71 6902.63"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M3697.65 6904.31V6843.96H3681.22V6903.8C3681.22 6912.82 3679.43 6919.57 3675.83 6924.04 3672.26 6928.54 3666.9 6930.79 3659.73 6930.79 3651.13 6930.79 3644.35 6928.07 3639.38 6922.64 3634.42 6917.24 3631.94 6909.86 3631.94 6900.5V6843.96H3615.52V6942.52H3631.94V6928.44C3635.78 6933.94 3640.3 6938.04 3645.51 6940.76 3650.71 6943.5 3656.71 6944.86 3663.51 6944.86 3674.75 6944.86 3683.25 6941.43 3688.99 6934.56 3694.76 6927.72 3697.65 6917.63 3697.65 6904.31"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M3783.08 6961.29C3770.49 6961.29 3760.48 6956.56 3753.05 6947.1 3745.64 6937.64 3741.94 6924.76 3741.94 6908.45 3741.94 6892.2 3745.64 6879.34 3753.05 6869.88 3760.48 6860.42 3770.49 6855.69 3783.08 6855.69 3795.69 6855.69 3805.68 6860.42 3813.04 6869.88 3820.39 6879.34 3824.07 6892.2 3824.07 6908.45 3824.07 6924.76 3820.39 6937.64 3813.04 6947.1 3805.68 6956.56 3795.69 6961.29 3783.08 6961.29M3783.08 6975.37C3801.22 6975.37 3815.71 6969.31 3826.57 6957.18 3837.42 6945.09 3842.84 6928.84 3842.84 6908.45 3842.84 6888.12 3837.42 6871.89 3826.57 6859.76 3815.71 6847.66 3801.22 6841.61 3783.08 6841.61 3764.89 6841.61 3750.36 6847.65 3739.48 6859.73 3728.61 6871.82 3723.17 6888.07 3723.17 6908.45 3723.17 6928.84 3728.61 6945.09 3739.48 6957.18 3750.36 6969.31 3764.89 6975.37 3783.08 6975.37"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M3971.78 6963.64V6944.86C3965.82 6950.36 3959.46 6954.47 3952.71 6957.18 3945.96 6959.92 3938.8 6961.29 3931.23 6961.29 3916.27 6961.29 3904.81 6956.76 3896.87 6947.69 3888.93 6938.64 3884.95 6925.57 3884.95 6908.45 3884.95 6891.39 3888.93 6878.33 3896.87 6869.26 3904.81 6860.21 3916.27 6855.69 3931.23 6855.69 3938.8 6855.69 3945.96 6857.05 3952.71 6859.76 3959.46 6862.5 3965.82 6866.62 3971.78 6872.12V6853.34C3965.57 6849.43 3959.01 6846.5 3952.09 6844.55 3945.17 6842.59 3937.85 6841.61 3930.13 6841.61 3910.3 6841.61 3894.68 6847.58 3883.27 6859.5 3871.88 6871.46 3866.18 6887.77 3866.18 6908.45 3866.18 6929.18 3871.88 6945.51 3883.27 6957.44 3894.68 6969.39 3910.3 6975.37 3930.13 6975.37 3937.95 6975.37 3945.32 6974.39 3952.23 6972.44 3959.18 6970.48 3965.69 6967.55 3971.78 6963.64"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M4018.72 6958.95V6858.04H4039.84C4057.69 6858.04 4070.75 6862.1 4079.04 6870.21 4087.32 6878.33 4091.47 6891.14 4091.47 6908.64 4091.47 6926.02 4087.32 6938.74 4079.04 6946.81 4070.75 6954.9 4057.69 6958.95 4039.84 6958.95H4018.72M3999.95 6973.02H4037.16C4062.34 6973.02 4080.81 6967.82 4092.57 6957.41 4104.35 6947.02 4110.24 6930.76 4110.24 6908.64 4110.24 6886.37 4104.33 6870.02 4092.5 6859.58 4080.66 6849.16 4062.22 6843.96 4037.16 6843.96H3999.95V6973.02"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M4137.07 6980.06H4151.3C4159.88 6966.3 4166.3 6952.82 4170.55 6939.62 4174.83 6926.45 4176.97 6913.36 4176.97 6900.35 4176.97 6887.3 4174.83 6874.15 4170.55 6860.9 4166.3 6847.68 4159.88 6834.21 4151.3 6820.49H4137.07C4144.95 6833.84 4150.82 6847.11 4154.71 6860.31 4158.6 6873.54 4160.54 6886.88 4160.54 6900.35 4160.54 6913.84


4158.6 6927.14 4154.71 6940.25 4150.82 6953.37 4144.95 6966.64 4137.07 6980.06"/> <path transform="matrix(.1,0,0,-.1,0,796)" stroke-width="30.3515" stroke-linecap="butt" stroke-miterlimit="10" stroke-linejoin="miter" fill="none" stroke="#000000" d="M1862.6 7026.34H2295.53"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1748.78 7026.34 1900.54 7102.21 1862.6 7026.34 1900.54 6950.46" fill-rule="evenodd"/> <path transform="matrix(.1,0,0,-.1,0,796)" stroke-width="2.8346" stroke-linecap="butt" stroke-miterlimit="10" stroke-linejoin="miter" fill="none" stroke="#000000" d="M1748.78 7026.34 1900.54 7102.21 1862.6 7026.34 1900.54 6950.46Z"/> <path transform="matrix(.1,0,0,-.1,0,796)" stroke-width="30.3515" stroke-linecap="butt" stroke-miterlimit="10" stroke-linejoin="miter" fill="none" stroke="#000000" d="M1748.78 7026.34 1900.54 7102.21 1862.6 7026.34 1900.54 6950.46Z"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M2409.35 7026.34 2257.59 6950.46 2295.53 7026.34 2257.59 7102.21" fill-rule="evenodd"/> <path transform="matrix(.1,0,0,-.1,0,796)" stroke-width="2.8346" stroke-linecap="butt" stroke-miterlimit="10" stroke-linejoin="miter" fill="none" stroke="#000000" d="M2409.35 7026.34 2257.59 6950.46 2295.53 7026.34 2257.59 7102.21Z"/> <path transform="matrix(.1,0,0,-.1,0,796)" stroke-width="30.3515" stroke-linecap="butt" stroke-miterlimit="10" stroke-linejoin="miter" fill="none" stroke="#000000" d="M2409.35 7026.34 2257.59 6950.46 2295.53 7026.34 2257.59 7102.21Z"/> <path transform="matrix(.1,0,0,-.1,0,796)" stroke-width="30.3515" stroke-linecap="butt" stroke-miterlimit="10" stroke-linejoin="miter" fill="none" stroke="#000000" d="M4715.64 7026.34H5027.16"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M4601.8 7026.34 4753.57 7102.21 4715.64 7026.34 4753.57 6950.46" fill-rule="evenodd"/> <path transform="matrix(.1,0,0,-.1,0,796)" stroke-width="2.8346" stroke-linecap="butt" stroke-miterlimit="10" stroke-linejoin="miter" fill="none" stroke="#000000" d="M4601.8 7026.34 4753.57 7102.21 4715.64 7026.34 4753.57 6950.46Z"/> <path transform="matrix(.1,0,0,-.1,0,796)" stroke-width="30.3515" stroke-linecap="butt" stroke-miterlimit="10" stroke-linejoin="miter" fill="none" stroke="#000000" d="M4601.8 7026.34 4753.57 7102.21 4715.64 7026.34 4753.57 6950.46Z"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M5140.97 7026.34 4989.2 6950.46 5027.16 7026.34 4989.2 7102.21" fill-rule="evenodd"/> <path transform="matrix(.1,0,0,-.1,0,796)" stroke-width="2.8346" stroke-linecap="butt" stroke-miterlimit="10" stroke-linejoin="miter" fill="none" stroke="#000000" d="M5140.97 7026.34 4989.2 6950.46 5027.16 7026.34 4989.2 7102.21Z"/> <path transform="matrix(.1,0,0,-.1,0,796)" stroke-width="30.3515" stroke-linecap="butt" stroke-miterlimit="10" stroke-linejoin="miter" fill="none" stroke="#000000" d="M5140.97 7026.34 4989.2 6950.46 5027.16 7026.34 4989.2 7102.21Z"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M3396.64 466.652H4307.183V1984.2219H3396.64Z" fill="#ffffff" fill-rule="evenodd"/> <path transform="matrix(.1,0,0,-.1,0,796)" stroke-width="2.8346" stroke-linecap="butt" stroke-miterlimit="10" stroke-linejoin="miter" fill="none" stroke="#ffffff" d="M3396.64 466.652H4307.183V1984.2219H3396.64Z"/> <path transform="matrix(.1,0,0,-.1,0,796)" stroke-width="30.3515" stroke-linecap="butt" stroke-dasharray="60.7027" stroke-miterlimit="10" stroke-linejoin="miter" fill="none" stroke="#000000" d="M3396.64 466.652H4307.183V1984.2219H3396.64Z"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M3609.74 1349.8V1333.37C3603.09 1336.52 3596.82 1338.87 3590.93 1340.41 3585.04 1341.97 3579.34 1342.75 3573.84 1342.75 3564.31 1342.75 3556.95 1340.95 3551.77 1337.33 3546.61 1333.71 3544.03 1328.57 3544.03 1321.89 3544.03 1316.29 3545.71 1312.07 3549.05 1309.21 3552.43 1306.37 3558.81 1304.07 3568.2 1302.31L3578.54 1300.19C3591.61 1297.69 3601.26 1293.28 3607.46 1286.95 3613.68 1280.64 3616.78 1272.18 3616.78 1261.58 3616.78 1248.94 3612.52 1239.36 3604.02 1232.83 3595.54 1226.33 3583.09 1223.08 3566.69 1223.08 3560.51 1223.08 3553.92 1223.86 3546.93 1225.42 3539.96 1227.01 3532.74 1229.36 3525.26 1232.46V1251.24C3532.42 1246.57 3539.43 1243.05 3546.27 1240.68 3553.14 1238.33 3559.88 1237.16 3566.51 1237.16 3576.58 1237.16 3584.34 1239.17 3589.79 1243.21 3595.27 1247.24 3598 1252.98 3598 1260.44 3598 1266.94 3596.1 1272.03 3592.29 1275.69 3588.47 1279.38 3582.21 1282.14 3573.51 1283.98L3563.1 1286C3549.78 1288.54 3540.13 1292.52 3534.17 1297.95 3528.23 1303.4 3525.26 1310.96 3525.26 1320.64 3525.26 1331.87 3529.34 1340.7 3537.5 1347.16 3545.7 1353.61 3556.96 1356.84 3571.31 1356.84 3577.47 1356.84 3583.74 1356.25 3590.12 1355.07 3596.53 1353.9 3603.07 1352.14 3609.74 1349.8"/> <path transform="matrix(.1,0,0,-.1,0,796)"

d="M3685.47 1216.7C3681.02 1204.77 3676.68 1196.98 3672.45 1193.34 3668.25 1189.7 3662.61 1187.88 3655.55 1187.88H3643.01V1201.96H3652.25C3656.57 1201.96 3659.92 1203 3662.29 1205.07 3664.69 1207.15 3667.34 1212.04 3670.25 1219.74L3673.07 1227.11 3633.62 1323.98H3651.04L3680.92 1246.91 3710.84 1323.98H3727.49L3685.47 1216.7"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M3813.98 1319.29V1305.21C3809.58 1307.55 3805.01 1309.32 3800.27 1310.49 3795.53 1311.66 3790.61 1312.25 3785.53 1312.25 3777.8 1312.25 3772.01 1311.04 3768.15 1308.62 3764.29 1306.2 3762.36 1302.57 3762.36 1297.73 3762.36 1294.04 3763.7 1291.14 3766.39 1289.04 3769.1 1286.94 3774.55 1284.93 3782.74 1283.03L3787.98 1281.82C3799.08 1279.37 3806.96 1275.91 3811.63 1271.44 3816.33 1266.96 3818.68 1260.72 3818.68 1252.7 3818.68 1243.59 3815.18 1236.36 3808.19 1231.04 3801.2 1225.73 3791.58 1223.08 3779.33 1223.08 3774.22 1223.08 3768.91 1223.66 3763.38 1224.84 3757.86 1226.01 3752.04 1227.77 3745.93 1230.12V1246.54C3751.67 1243.41 3757.33 1241.07 3762.91 1239.5 3768.48 1237.94 3773.99 1237.16 3779.44 1237.16 3786.77 1237.16 3792.41 1238.44 3796.34 1241.01 3800.28 1243.6 3802.25 1247.23 3802.25 1251.89 3802.25 1256.25 3800.86 1259.57 3798.07 1261.87 3795.28 1264.19 3789.16 1266.43 3779.7 1268.58L3774.38 1269.86C3764.34 1271.96 3757.07 1275.19 3752.6 1279.54 3748.15 1283.89 3745.93 1289.86 3745.93 1297.44 3745.93 1306.65 3749.16 1313.77 3755.61 1318.78 3762.06 1323.81 3771.21 1326.33 3783.07 1326.33 3788.94 1326.33 3794.46 1325.74 3799.64 1324.57 3804.83 1323.39 3809.61 1321.64 3813.98 1319.29"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M3863.55 1352.14V1323.98H3896.41V1312.25H3863.55V1258.5C3863.55 1250.43 3864.63 1245.25 3866.78 1242.95 3868.96 1240.65 3873.37 1239.5 3880.02 1239.5H3896.41V1225.42H3880.02C3867.58 1225.42 3858.98 1227.79 3854.24 1232.54 3849.5 1237.3 3847.13 1245.96 3847.13 1258.5V1312.25H3835.39V1323.98H3847.13V1352.14H3863.55"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M4002.41 1279.69V1272.36H3927.32C3928.02 1260.89 3931.37 1252.15 3937.36 1246.14 3943.35 1240.15 3951.69 1237.16 3962.37 1237.16 3968.58 1237.16 3974.59 1237.94 3980.41 1239.5 3986.23 1241.07 3992 1243.41 3997.71 1246.54V1232.46C3991.95 1229.41 3986.03 1227.07 3979.97 1225.46 3973.93 1223.87 3967.81 1223.08 3961.6 1223.08 3946 1223.08 3933.64 1227.64 3924.53 1236.75 3915.43 1245.89 3910.89 1258.25 3910.89 1273.82 3910.89 1289.93 3915.25 1302.7 3923.98 1312.14 3932.7 1321.6 3944.48 1326.33 3959.29 1326.33 3972.59 1326.33 3983.1 1322.15 3990.82 1313.79 3998.55 1305.43 4002.41 1294.06 4002.41 1279.69M3985.98 1284.09C3985.86 1292.64 3983.35 1299.48 3978.46 1304.59 3973.6 1309.7 3967.16 1312.25 3959.14 1312.25 3950.05 1312.25 3942.76 1309.78 3937.29 1304.84 3931.84 1299.91 3928.7 1292.95 3927.86 1283.98L3985.98 1284.09"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M4106.82 1306.23C4110.86 1313.11 4115.66 1318.16 4121.23 1321.41 4126.83 1324.69 4133.43 1326.33 4141.04 1326.33 4151.23 1326.33 4159.09 1322.81 4164.61 1315.77 4170.16 1308.75 4172.93 1298.76 4172.93 1285.78V1225.42H4156.51V1285.26C4156.51 1294.41 4154.86 1301.19 4151.56 1305.61 4148.26 1310.04 4143.21 1312.25 4136.41 1312.25 4128.13 1312.25 4121.58 1309.54 4116.76 1304.11 4111.97 1298.71 4109.57 1291.32 4109.57 1281.96V1225.42H4093.15V1285.26C4093.15 1294.45 4091.5 1301.25 4088.2 1305.65 4084.9 1310.05 4079.8 1312.25 4072.91 1312.25 4064.72 1312.25 4058.22 1309.52 4053.4 1304.07 4048.61 1298.62 4046.21 1291.25 4046.21 1281.96V1225.42H4029.79V1323.98H4046.21V1309.9C4049.88 1315.52 4054.27 1319.67 4059.38 1322.33 4064.51 1325 4070.6 1326.33 4077.64 1326.33 4084.75 1326.33 4090.79 1324.61 4095.75 1321.16 4100.74 1317.74 4104.43 1312.76 4106.82 1306.23"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M3720.22 1062.02V1012.73H3747.21C3756.37 1012.73 3763.16 1014.77 3767.55 1018.82 3771.98 1022.88 3774.19 1029.08 3774.19 1037.41 3774.19 1045.82 3771.98 1052.02 3767.55 1056 3763.16 1060.01 3756.37 1062.02 3747.21 1062.02H3720.22M3720.22 1113.64V1076.09H3745.12C3753.33 1076.09 3759.44 1077.64 3763.45 1080.71 3767.48 1083.82 3769.5 1088.54 3769.5 1094.87 3769.5 1101.15 3767.48 1105.84 3763.45 1108.95 3759.44 1112.08 3753.33 1113.64 3745.12 1113.64H3720.22M3701.45 1127.72H3746.36C3759.78 1127.72 3770.12 1125.03 3777.38 1119.66 3784.64 1114.28 3788.27 1106.63 3788.27 1096.7 3788.27 1089.03 3786.34 1082.91 3782.48 1078.37 3778.62 1073.85 3772.93 1071.02 3765.43 1069.9 3774.13 1068.02 3780.89 1064.08 3785.71 1058.09 3790.55 1052.1 3792.96 1044.63 3792.96 1035.65 3792.96 1023.84 3789.07 1014.73 3781.27 1008.3 3773.47


1001.87 3762.39 998.656 3748.01 998.656H3701.45V1127.72"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M3826.05 1036.79V1097.21H3842.48V1037.41C3842.48 1028.42 3844.26 1021.66 3847.83 1017.14 3851.4 1012.64 3856.77 1010.39 3863.93 1010.39 3872.51 1010.39 3879.29 1013.09 3884.28 1018.49 3889.27 1023.92 3891.76 1031.3 3891.76 1040.64V1097.21H3908.18V998.656H3891.76V1012.73C3887.87 1007.19 3883.35 1003.05 3878.19 1000.34 3873.06 997.652 3867.09 996.309 3860.3 996.309 3849.08 996.309 3840.56 999.742 3834.74 1006.61 3828.95 1013.5 3826.05 1023.56 3826.05 1036.79M3866.61 1099.56V1099.56"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M4001.76 1092.52V1078.44C3997.36 1080.79 3992.79 1082.55 3988.04 1083.72 3983.3 1084.89 3978.39 1085.48 3973.3 1085.48 3965.58 1085.48 3959.79 1084.27 3955.92 1081.85 3952.06 1079.43 3950.13 1075.8 3950.13 1070.96 3950.13 1067.27 3951.48 1064.38 3954.16 1062.27 3956.88 1060.17 3962.33 1058.16 3970.52 1056.26L3975.76 1055.05C3986.86 1052.61 3994.74 1049.14 3999.41 1044.67 4004.1 1040.2 4006.45 1033.95 4006.45 1025.94 4006.45 1016.82 4002.95 1009.59 3995.96 1004.27 3988.97 998.961 3979.35 996.309 3967.11 996.309 3962 996.309 3956.68 996.895 3951.16 998.07 3945.63 999.242 3939.82 1001 3933.7 1003.35V1019.78C3939.45 1016.65 3945.11 1014.3 3950.68 1012.73 3956.25 1011.17 3961.77 1010.39 3967.22 1010.39 3974.55 1010.39 3980.18 1011.67 3984.12 1014.24 3988.05 1016.83 3990.02 1020.46 3990.02 1025.13 3990.02 1029.48 3988.63 1032.8 3985.84 1035.1 3983.06 1037.43 3976.93 1039.66 3967.47 1041.81L3962.16 1043.09C3952.11 1045.2 3944.85 1048.43 3940.38 1052.77 3935.93 1057.13 3933.7 1063.09 3933.7 1070.67 3933.7 1079.88 3936.93 1087 3943.38 1092.01 3949.84 1097.04 3958.99 1099.56 3970.85 1099.56 3976.71 1099.56 3982.24 1098.98 3987.42 1097.8 3992.6 1096.63 3997.38 1094.87 4001.76 1092.52"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M5252.89 1346.54H6628.78V2256.30H5252.89Z" fill="#ffffff" fill-rule="evenodd"/> <path transform="matrix(.1,0,0,-.1,0,796)" stroke-width="2.8346" stroke-linecap="butt" stroke-miterlimit="10" stroke-linejoin="miter" fill="none" stroke="#ffffff" d="M5252.89 1346.54H6628.78V2256.30H5252.89Z"/> <path transform="matrix(.1,0,0,-.1,0,796)" stroke-width="27.5177" stroke-linecap="butt" stroke-dasharray="55.0355" stroke-miterlimit="10" stroke-linejoin="miter" fill="none" stroke="#000000" d="M5252.89 1346.54H6628.78V2256.30H5252.89Z"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M5525.91 2113.29H5544.68V2061.66H5605.7V2113.29H5624.47V1984.22H5605.7V2047.58H5544.68V1984.22 H5525.91V2113.29"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M5705.59 2033.5C5692.93 2033.5 5684.16 2032.04 5679.27 2029.13 5674.38 2026.23 5671.93 2021.25 5671.93 2014.21 5671.93 2008.61 5673.77 2004.16 5677.43 2000.86 5681.1 1997.59 5686.09 1995.95 5692.39 1995.95 5701.1 1995.95 5708.07 1999.03 5713.33 2005.19 5718.59 2011.38 5721.21 2019.59 5721.21 2029.83V2033.5H5705.59M5737.64 2039.7V1984.22H5721.21V1998.3C5717.6 1992.68 5713.09 1988.53 5707.68 1985.87 5702.28 1983.2 5695.67 1981.87 5687.85 1981.87 5677.95 1981.87 5670.08 1984.63 5664.23 1990.16 5658.42 1995.71 5655.51 2003.13 5655.51 2012.41 5655.51 2023.27 5659.16 2031.45 5666.47 2036.95 5673.78 2042.47 5684.7 2045.23 5699.21 2045.23H5721.21V2046.99C5721.21 2054.64 5718.81 2060.56 5713.99 2064.74 5709.2 2068.94 5702.46 2071.04 5693.79 2071.04 5688.27 2071.04 5682.89 2070.46 5677.66 2069.29 5672.43 2068.11 5667.39 2066.35 5662.55 2064V2078.08C5668.39 2080.43 5674.06 2082.19 5679.56 2083.36 5685.09 2084.54 5690.45 2085.13 5695.66 2085.13 5709.74 2085.13 5720.25 2081.36 5727.19 2073.83 5734.16 2066.32 5737.64 2054.95 5737.64 2039.7"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M5830.02 2068.7C5828.26 2069.5 5826.36 2070.09 5824.3 2070.46 5822.25 2070.85 5819.97 2071.04 5817.48 2071.04 5808.68 2071.04 5801.92 2068.1 5797.2 2062.21 5792.49 2056.34 5790.13 2047.91 5790.13 2036.91V1984.22H5773.7V2082.78H5790.13V2068.7C5793.4 2074.25 5797.67 2078.36 5802.93 2081.05 5808.18 2083.77 5814.56 2085.13 5822.06 2085.13 5823.14 2085.13 5824.32 2085.13 5825.62 2085.13 5826.92 2085.13 5828.36 2085.13 5829.95 2085.13L5830.02 2068.7"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M5908.57 2068.7V2120.32H5924.99V1984.22H5908.57V1998.3C5905.24 1992.75 5901.04 1988.62 5895.95 1985.91 5890.87 1983.21 5884.76 1981.87 5877.62 1981.87 5865.96 1981.87 5856.46 1986.61 5849.13

1996.1 5841.82 2005.58 5838.17 2018.05 5838.17 2033.5 5838.17 2048.95 5841.82 2061.41 5849.13 2070.9 5856.46 2080.38 5865.96 2085.13 5877.62 2085.13 5884.76 2085.13 5890.87 2083.77 5895.95 2081.05 5901.04 2078.36 5905.24 2074.25 5908.57 2068.7M5854.59 2033.5C5854.59 2021.79 5856.99 2012.6 5861.78 2005.93 5866.57 1999.28 5873.16 1995.95 5881.54 1995.95 5889.93 1995.95 5896.53 1999.28 5901.34 2005.93 5906.16 2012.6 5908.57 2021.79 5908.57 2033.5 5908.57 2045.21 5906.16 2054.39 5901.34 2061.04 5896.53 2067.71 5889.93 2071.04 5881.54 2071.04 5873.16 2071.04 5866.57 2067.71 5861.78 2061.04 5856.99 2054.39 5854.59 2045.21 5854.59 2033.5"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M5951.22 2082.78H5967.32L5987.41 2005.85 6007.43 2082.78H6026.43L6046.52 2005.85 6066.54 2082.78H6082.64L6057.01 1984.22H6038.05L6016.96 2064.99 5995.81 1984.22H5976.86L5951.22 2082.78"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6151.63 2033.5C6138.96 2033.5 6130.19 2032.04 6125.3 2029.13 6120.41 2026.23 6117.96 2021.25 6117.96 2014.21 6117.96 2008.61 6119.8 2004.16 6123.46 2000.86 6127.13 1997.59 6132.12 1995.95 6138.43 1995.95 6147.13 1995.95 6154.11 1999.03 6159.36 2005.19 6164.62 2011.38 6167.25 2019.59 6167.25 2029.83V2033.5H6151.63M6183.67 2039.7V1984.22H6167.25V1998.3C6163.63 1992.68 6159.12 1988.53 6153.71 1985.87 6148.31 1983.2 6141.7 1981.87 6133.88 1981.87 6123.98 1981.87 6116.11 1984.63 6110.27 1990.16 6104.45 1995.71 6101.54 2003.13 6101.54 2012.41 6101.54 2023.27 6105.19 2031.45 6112.5 2036.95 6119.81 2042.47 6130.72 2045.23 6145.25 2045.23H6167.25V2046.99C6167.25 2054.64 6164.84 2060.56 6160.02 2064.74 6155.23 2068.94 6148.5 2071.04 6139.82 2071.04 6134.29 2071.04 6128.91 2070.46 6123.68 2069.29 6118.45 2068.11 6113.42 2066.35 6108.58 2064V2078.08C6114.42 2080.43 6120.09 2082.19 6125.59 2083.36 6131.12 2084.54 6136.48 2085.13 6141.69 2085.13 6155.77 2085.13 6166.28 2081.36 6173.22 2073.83 6180.19 2066.32 6183.67 2054.95 6183.67 2039.7"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6276.05 2068.7C6274.29 2069.5 6272.38 2070.09 6270.33 2070.46 6268.28 2070.85 6266 2071.04 6263.51 2071.04 6254.71 2071.04 6247.95 2068.1 6243.23 2062.21 6238.52 2056.34 6236.16 2047.91 6236.16 2036.91V1984.22H6219.73V2082.78H6236.16V2068.7C6239.43 2074.25 6243.7 2078.36 6248.95 2081.05 6254.21 2083.77 6260.59 2085.13 6268.09 2085.13 6269.17 2085.13 6270.36 2085.13 6271.65 2085.13 6272.95 2085.13 6274.39 2085.13 6275.98 2085.13L6276.05 2068.7"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6374.3 2038.48V2031.15H6299.21C6299.91 2019.69 6303.27 2010.95 6309.25 2004.93 6315.24 1998.95 6323.58 1995.95 6334.26 1995.95 6340.47 1995.95 6346.48 1996.73 6352.3 1998.3 6358.12 1999.86 6363.89 2002.21 6369.61 2005.34V1991.26C6363.84 1988.2 6357.92 1985.87 6351.86 1984.25 6345.82 1982.66 6339.7 1981.87 6333.49 1981.87 6317.89 1981.87 6305.54 1986.43 6296.42 1995.55 6287.33 2004.69 6282.78 2017.05 6282.78 2032.62 6282.78 2048.73 6287.14 2061.5 6295.87 2070.93 6304.6 2080.39 6316.37 2085.13 6331.18 2085.13 6344.48 2085.13 6354.99 2080.95 6362.71 2072.58 6370.44 2064.22 6374.3 2052.86 6374.3 2038.48M6357.87 2042.88C6357.75 2051.44 6355.25 2058.27 6350.36 2063.38 6345.49 2068.49 6339.05 2071.04 6331.03 2071.04 6321.94 2071.04 6314.66 2068.57 6309.18 2063.64 6303.73 2058.7 6300.59 2051.75 6299.76 2042.77L6357.87 2042.88"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M5632.57 1886.52H5745.2V1872.44H5698.27V1757.45H5679.5V1872.44H5632.57V1886.52"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M5791.2 1841.93C5789.44 1842.73 5787.53 1843.32 5785.48 1843.69 5783.43 1844.08 5781.15 1844.28 5778.66 1844.28 5769.86 1844.28 5763.1 1841.33 5758.38 1835.44 5753.66 1829.57 5751.31 1821.14 5751.31 1810.14V1757.45H5734.88V1856.01H5751.31V1841.93C5754.58 1847.48 5758.85 1851.6 5764.1 1854.29 5769.36 1857 5775.74 1858.36 5783.24 1858.36 5784.32 1858.36 5785.5 1858.36 5786.8 1858.36 5788.09 1858.36 5789.54 1858.36 5791.13 1858.36L5791.2 1841.93"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M5809.22 1856.01H5825.64V1757.45H5809.22V1856.01M5809.22 1893.55H5825.64V1872.44H5809.22V1893.55"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M5922.84 1807.87C5922.84 1819.43 5920.47 1828.39 5915.73 1834.74 5911.01 1841.1 5904.38 1844.28 5895.82 1844.28 5887.31 1844.28 5880.69 1841.1 5875.95 1834.74 5871.23 1828.39 5868.87 1819.43 5868.87 1807.87 5868.87 1796.35 5871.23 1787.42 5875.95 1781.06 5880.69 1774.71 5887.31


1771.53 5895.82 1771.53 5904.38 1771.53 5911.01 1774.71 5915.73 1781.06 5920.47 1787.42 5922.84 1796.35 5922.84 1807.87M5939.27 1770.17C5939.27 1753.21 5935.59 1740.6 5928.23 1732.33 5920.9 1724.05 5909.67 1719.9 5894.54 1719.9 5888.91 1719.9 5883.62 1720.3 5878.66 1721.08 5873.7 1721.84 5868.87 1723.01 5864.18 1724.6V1741.02C5868.82 1738.63 5873.41 1736.86 5877.93 1735.71 5882.47 1734.56 5887.09 1733.98 5891.79 1733.98 5902.18 1733.98 5909.95 1736.73 5915.11 1742.23 5920.27 1747.73 5922.84 1756.03 5922.84 1767.13V1773.88C5919.57 1768.38 5915.38 1764.26 5910.27 1761.52 5905.16 1758.81 5899.04 1757.45 5891.9 1757.45 5880.07 1757.45 5870.53 1762.05 5863.3 1771.24 5856.06 1780.45 5852.45 1792.66 5852.45 1807.87 5852.45 1823.12 5856.06 1835.34 5863.3 1844.53 5870.53 1853.75 5880.07 1858.36 5891.9 1858.36 5899.04 1858.36 5905.16 1856.99 5910.27 1854.25 5915.38 1851.54 5919.57 1847.43 5922.84 1841.93V1856.01H5939.27V1770.17"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6038.25 1807.87C6038.25 1819.43 6035.88 1828.39 6031.13 1834.74 6026.41 1841.1 6019.78 1844.28 6011.22 1844.28 6002.71 1844.28 5996.09 1841.1 5991.35 1834.74 5986.63 1828.39 5984.27 1819.43 5984.27 1807.87 5984.27 1796.35 5986.63 1787.42 5991.35 1781.06 5996.09 1774.71 6002.71 1771.53 6011.22 1771.53 6019.78 1771.53 6026.41 1774.71 6031.13 1781.06 6035.88 1787.42 6038.25 1796.35 6038.25 1807.87M6054.67 1770.17C6054.67 1753.21 6050.99 1740.6 6043.64 1732.33 6036.3 1724.05 6025.07 1719.9 6009.94 1719.9 6004.32 1719.9 5999.02 1720.3 5994.06 1721.08 5989.1 1721.84 5984.27 1723.01 5979.58 1724.6V1741.02C5984.22 1738.63 5988.81 1736.86 5993.33 1735.71 5997.88 1734.56 6002.5 1733.98 6007.19 1733.98 6017.58 1733.98 6025.35 1736.73 6030.51 1742.23 6035.67 1747.73 6038.25 1756.03 6038.25 1767.13V1773.88C6034.97 1768.38 6030.78 1764.26 6025.67 1761.52 6020.56 1758.81 6014.44 1757.45 6007.3 1757.45 5995.47 1757.45 5985.93 1762.05 5978.7 1771.24 5971.46 1780.45 5967.85 1792.66 5967.85 1807.87 5967.85 1823.12 5971.46 1835.34 5978.7 1844.53 5985.93 1853.75 5995.47 1858.36 6007.3 1858.36 6014.44 1858.36 6020.56 1856.99 6025.67 1854.25 6030.78 1851.54 6034.97 1847.43 6038.25 1841.93V1856.01H6054.67V1770.17"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6174.77 1811.71V1804.38H6099.68C6100.38 1792.92 6103.73 1784.18 6109.72 1778.17 6115.71 1772.18 6124.05 1769.18 6134.73 1769.18 6140.94 1769.18 6146.95 1769.96 6152.77 1771.53 6158.59 1773.09 6164.36 1775.44 6170.07 1778.57V1764.49C6164.3 1761.43 6158.39 1759.1 6152.33 1757.49 6146.29 1755.9 6140.17 1755.11 6133.96 1755.11 6118.36 1755.11 6106 1759.66 6096.89 1768.78 6087.79 1777.92 6083.25 1790.28 6083.25 1805.85 6083.25 1821.96 6087.61 1834.73 6096.34 1844.17 6105.07 1853.63 6116.84 1858.36 6131.65 1858.36 6144.95 1858.36 6155.46 1854.18 6163.18 1845.82 6170.91 1837.46 6174.77 1826.09 6174.77 1811.71M6158.34 1816.12C6158.22 1824.67 6155.71 1831.5 6150.82 1836.61 6145.96 1841.72 6139.52 1844.28 6131.5 1844.28 6122.41 1844.28 6115.13 1841.81 6109.65 1836.87 6104.2 1831.93 6101.05 1824.98 6100.23 1816.01L6158.34 1816.12"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6258.47 1841.93C6256.71 1842.73 6254.8 1843.32 6252.75 1843.69 6250.7 1844.08 6248.42 1844.28 6245.93 1844.28 6237.13 1844.28 6230.37 1841.33 6225.65 1835.44 6220.93 1829.57 6218.57 1821.14 6218.57 1810.14V1757.45H6202.15V1856.01H6218.57V1841.93C6221.85 1847.48 6226.12 1851.6 6231.37 1854.29 6236.63 1857 6243.01 1858.36 6250.51 1858.36 6251.59 1858.36 6252.77 1858.36 6254.07 1858.36 6255.36 1858.36 6256.8 1858.36 6258.39 1858.36L6258.47 1841.93"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M5633.98 1659.75H5661.62L5693.78 1572.77 5726.12 1659.75H5753.66V1530.68H5734.88V1644.16L5702.39 1556.5H5685.23L5652.75 1644.16V1530.68H5633.98V1659.75"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M5828.02 1617.51C5819.32 1617.51 5812.44 1614.16 5807.38 1607.46 5802.34 1600.76 5799.83 1591.6 5799.83 1579.96 5799.83 1568.32 5802.33 1559.16 5807.34 1552.46 5812.38 1545.77 5819.27 1542.41 5828.02 1542.41 5836.68 1542.41 5843.52 1545.78 5848.56 1552.5 5853.62 1559.22 5856.15 1568.38 5856.15 1579.96 5856.15 1591.5 5853.62 1600.64 5848.56 1607.39 5843.52 1614.13 5836.68 1617.51 5828.02 1617.51M5827.99 1631.59C5841.87 1631.59 5852.77 1627.02 5860.7 1617.88 5868.61 1608.76 5872.57 1596.12 5872.57 1579.96 5872.57 1563.85 5868.61 1551.21 5860.7 1542.05 5852.77 1532.91 5841.87 1528.34 5827.99 1528.34 5814.05 1528.34 5803.14 1532.91 5795.25 1542.05 5787.35 1551.21 5783.4 1563.85 5783.4 1579.96 5783.4 1596.12 5787.35 1608.76 5795.25 1617.88 5803.14 1627.02 5814.05 1631.59 5827.99 1631.59"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M5964.95 1615.16V1666.79H5981.38V1530.68H5964.95V1544.76C5961.63 1539.21 5957.43 1535.08 5952.34


1532.37 5947.25 1529.68 5941.14 1528.34 5934.01 1528.34 5922.35 1528.34 5912.85 1533.08 5905.52 1542.56 5898.21 1552.05 5894.55 1564.51 5894.55 1579.96 5894.55 1595.41 5898.21 1607.88 5905.52 1617.36 5912.85 1626.84 5922.35 1631.59 5934.01 1631.59 5941.14 1631.59 5947.25 1630.23 5952.34 1627.52 5957.43 1624.83 5961.63 1620.71 5964.95 1615.16M5910.98 1579.96C5910.98 1568.25 5913.38 1559.06 5918.17 1552.39 5922.96 1545.74 5929.55 1542.41 5937.93 1542.41 5946.31 1542.41 5952.91 1545.74 5957.73 1552.39 5962.55 1559.06 5964.95 1568.25 5964.95 1579.96 5964.95 1591.67 5962.55 1600.85 5957.73 1607.5 5952.91 1614.17 5946.31 1617.51 5937.93 1617.51 5929.55 1617.51 5922.96 1614.17 5918.17 1607.5 5913.38 1600.85 5910.98 1591.67 5910.98 1579.96"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6017 1568.82V1629.24H6033.42V1569.44C6033.42 1560.44 6035.21 1553.68 6038.78 1549.16 6042.34 1544.66 6047.71 1542.41 6054.87 1542.41 6063.45 1542.41 6070.23 1545.12 6075.22 1550.52 6080.21 1555.95 6082.7 1563.33 6082.7 1572.66V1629.24H6099.13V1530.68H6082.7V1544.76C6078.82 1539.21 6074.29 1535.08 6069.14 1532.37 6064 1529.68 6058.04 1528.34 6051.24 1528.34 6040.02 1528.34 6031.5 1531.77 6025.69 1538.64 6019.89 1545.53 6017 1555.59 6017 1568.82M6057.55 1631.59V1631.59"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6131.69 1666.79H6148.11V1530.68H6131.69V1666.79"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6266.43 1584.95V1577.61H6191.34C6192.05 1566.15 6195.4 1557.41 6201.39 1551.4 6207.38 1545.41 6215.71 1542.41 6226.39 1542.41 6232.6 1542.41 6238.61 1543.2 6244.43 1544.76 6250.25 1546.32 6256.02 1548.67 6261.74 1551.8V1537.72C6255.97 1534.67 6250.05 1532.33 6243.99 1530.72 6237.95 1529.13 6231.83 1528.34 6225.62 1528.34 6210.03 1528.34 6197.67 1532.89 6188.55 1542.01 6179.46 1551.15 6174.91 1563.51 6174.91 1579.08 6174.91 1595.19 6179.28 1607.96 6188 1617.4 6196.73 1626.86 6208.5 1631.59 6223.31 1631.59 6236.61 1631.59 6247.12 1627.41 6254.85 1619.05 6262.57 1610.69 6266.43 1599.32 6266.43 1584.95M6250 1589.35C6249.88 1597.9 6247.38 1604.73 6242.49 1609.84 6237.63 1614.95 6231.18 1617.51 6223.17 1617.51 6214.07 1617.51 6206.79 1615.04 6201.31 1610.1 6195.86 1605.16 6192.72 1598.21 6191.89 1589.24L6250 1589.35"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M397.152 5872.12C400.941 5870.88 404.633 5868.2 408.227 5864.09 411.844 5859.99 415.461 5854.35 419.078 5847.19L436.02 5812.29H418.016L401.188 5845.03C396.957 5853.58 392.863 5859.25 388.902 5862.04 384.945 5864.85 379.543 5866.26 372.695 5866.26H353.887V5812.29H335.113V5941.35H375.594C390.383 5941.35 401.406 5938.23 408.668 5932 415.949 5925.77 419.594 5916.37 419.594 5903.8 419.594 5895.61 417.676 5888.8 413.836 5883.38 410.023 5877.98 404.461 5874.23 397.152 5872.12M353.887 5927.27V5880.34H375.594C383.93 5880.34 390.211 5882.33 394.441 5886.31 398.695 5890.3 400.82 5896.15 400.82 5903.88 400.82 5911.63 398.695 5917.45 394.441 5921.37 390.211 5925.3 383.93 5927.27 375.594 5927.27H353.887"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M461.137 5941.35H479.91V5812.29H461.137V5941.35"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M594.73 5936.66V5920.23C588.078 5923.38 581.809 5925.73 575.918 5927.27 570.027 5928.83 564.332 5929.62 558.832 5929.62 549.301 5929.62 541.941 5927.81 536.758 5924.19 531.602 5920.57 529.023 5915.43 529.023 5908.75 529.023 5903.16 530.695 5898.93 534.047 5896.07 537.418 5893.23 543.801 5890.93 553.188 5889.17L563.527 5887.05C576.602 5884.55 586.246 5880.14 592.457 5873.81 598.664 5867.5 601.77 5859.05 601.77 5848.44 601.77 5835.8 597.516 5826.21 589.008 5819.69 580.527 5813.19 568.086 5809.94 551.684 5809.94 545.496 5809.94 538.91 5810.72 531.918 5812.29 524.953 5813.87 517.73 5816.22 510.25 5819.32V5838.1C517.41 5833.43 524.414 5829.91 531.258 5827.54 538.129 5825.19 544.875 5824.02 551.5 5824.02 561.57 5824.02 569.332 5826.04 574.781 5830.07 580.258 5834.1 582.996 5839.84 582.996 5847.3 582.996 5853.8 581.09 5858.89 577.273 5862.55 573.461 5866.24 567.203 5869.01 558.504 5870.84L548.09 5872.86C534.766 5875.4 525.125 5879.38 519.16 5884.81 513.219 5890.26 510.25 5897.82 510.25 5907.51 510.25 5918.73 514.332 5927.56 522.496 5934.02 530.684 5940.47 541.953 5943.7 556.301 5943.7 562.461 5943.7 568.73 5943.11 575.113 5941.93 581.516 5940.76 588.055 5939 594.73 5936.66"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M728.902 5931.96V5913.19C722.938 5918.69 716.582 5922.8 709.836 5925.51 703.09 5928.25 695.93 5929.62 688.352 5929.62 673.391 5929.62 661.938 5925.08 653.996 5916.01 646.051 5906.97 642.078 5893.89 642.078 5876.78 642.078 5859.72 646.051 5846.65 653.996 5837.58 661.938 5828.54 673.391 5824.02 688.352

5824.02 695.93 5824.02 703.09 5825.38 709.836 5828.09 716.582 5830.82 722.938 5834.94 728.902 5840.44V5821.67C722.695 5817.76 716.133 5814.82 709.215 5812.87 702.297 5810.91 694.977 5809.94 687.25 5809.94 667.426 5809.94 651.805 5815.9 640.391 5827.83 629 5839.78 623.305 5856.1 623.305 5876.78 623.305 5897.51 629 5913.84 640.391 5925.77 651.805 5937.72 667.426 5943.7 687.25 5943.7 695.074 5943.7 702.441 5942.72 709.359 5940.76 716.301 5938.8 722.816 5935.88 728.902 5931.96"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M750.035 5868.6H796.969V5854.52H750.035V5868.6"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M847.375 5812.29 797.508 5941.35H815.953L857.348 5832.67 898.816 5941.35H917.188L867.395 5812.29H847.375"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1011.61 5927.27V5877.99H1033.31C1041.36 5877.99 1047.56 5880.13 1051.94 5884.41 1056.34 5888.68 1058.54 5894.77 1058.54 5902.66 1058.54 5910.51 1056.34 5916.57 1051.94 5920.85 1047.56 5925.13 1041.36 5927.27 1033.31 5927.27H1011.61M992.832 5941.35H1033.31C1047.81 5941.35 1058.76 5938.06 1066.16 5931.48 1073.6 5924.93 1077.31 5915.33 1077.31 5902.66 1077.31 5889.91 1073.6 5880.25 1066.16 5873.7 1058.76 5867.17 1047.81 5863.91 1033.31 5863.91H1011.61V5812.29H992.832V5941.35"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1101.86 5948.39H1118.29V5812.29H1101.86V5948.39"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1195.17 5861.56C1182.51 5861.56 1173.73 5860.11 1168.85 5857.2 1163.96 5854.29 1161.52 5849.32 1161.52 5842.28 1161.52 5836.68 1163.35 5832.23 1167.02 5828.93 1170.68 5825.66 1175.67 5824.02 1181.97 5824.02 1190.68 5824.02 1197.66 5827.1 1202.91 5833.26 1208.16 5839.44 1210.79 5847.65 1210.79 5857.89V5861.56H1195.17M1227.22 5867.76V5812.29H1210.79V5826.36C1207.18 5820.74 1202.66 5816.6 1197.27 5813.93 1191.86 5811.27 1185.25 5809.94 1177.43 5809.94 1167.53 5809.94 1159.66 5812.7 1153.81 5818.22 1148 5823.77 1145.09 5831.19 1145.09 5840.48 1145.09 5851.33 1148.74 5859.51 1156.05 5865.01 1163.36 5870.54 1174.27 5873.3 1188.79 5873.3H1210.79V5875.05C1210.79 5882.71 1208.39 5888.62 1203.57 5892.8 1198.78 5897.01 1192.04 5899.11 1183.37 5899.11 1177.84 5899.11 1172.46 5898.52 1167.23 5897.35 1162 5896.18 1156.97 5894.41 1152.13 5892.07V5906.15C1157.97 5908.5 1163.64 5910.25 1169.14 5911.43 1174.66 5912.6 1180.03 5913.19 1185.24 5913.19 1199.32 5913.19 1209.83 5909.43 1216.77 5901.89 1223.74 5894.39 1227.22 5883.01 1227.22 5867.76"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1279.71 5939V5910.84H1312.56V5899.11H1279.71V5845.36C1279.71 5837.29 1280.78 5832.11 1282.93 5829.81 1285.11 5827.51 1289.52 5826.36 1296.17 5826.36H1312.56V5812.29H1296.17C1283.73 5812.29 1275.14 5814.66 1270.39 5819.4 1265.65 5824.16 1263.28 5832.82 1263.28 5845.36V5899.11H1251.55V5910.84H1263.28V5939H1279.71"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1383.36 5948.39V5934.31H1367.85C1362.23 5934.31 1358.32 5933.2 1356.12 5930.97 1353.94 5928.77 1352.85 5924.8 1352.85 5919.05V5910.84H1378.66V5899.11H1352.85V5812.29H1336.43V5899.11H1320V5910.84H1336.43V5917.3C1336.43 5928.13 1338.95 5936.01 1343.98 5940.95 1349.04 5945.91 1357.06 5948.39 1368.03 5948.39H1383.36"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1435.38 5899.11C1426.68 5899.11 1419.8 5895.76 1414.74 5889.06 1409.7 5882.36 1407.18 5873.2 1407.18 5861.56 1407.18 5849.93 1409.69 5840.76 1414.7 5834.06 1419.74 5827.37 1426.63 5824.02 1435.38 5824.02 1444.04 5824.02 1450.88 5827.38 1455.91 5834.1 1460.97 5840.82 1463.5 5849.98 1463.5 5861.56 1463.5 5873.1 1460.97 5882.24 1455.91 5888.99 1450.88 5895.73 1444.04 5899.11 1435.38 5899.11M1435.34 5913.19C1449.23 5913.19 1460.13 5908.62 1468.05 5899.48 1475.97 5890.36 1479.93 5877.72 1479.93 5861.56 1479.93 5845.45 1475.97 5832.82 1468.05 5823.65 1460.13 5814.51 1449.23 5809.94 1435.34 5809.94 1421.41 5809.94 1410.5 5814.51 1402.6 5823.65 1394.71 5832.82 1390.76 5845.45 1390.76 5861.56 1390.76 5877.72 1394.71 5890.36 1402.6 5899.48 1410.5 5908.62 1421.41 5913.19 1435.34 5913.19"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1565.27 5896.76C1563.51 5897.57 1561.61 5898.16 1559.55 5898.52 1557.5 5898.91 1555.22 5899.11 1552.73 5899.11 1543.93 5899.11 1537.17 5896.16 1532.45 5890.27 1527.73 5884.41 1525.38 5875.97 1525.38 5864.97V5812.29H1508.95V5910.84H1525.38V5896.76C1528.65 5902.31 1532.92 5906.43 1538.17 5909.12 1543.43 5911.83 1549.81 5913.19 1557.31 5913.19 1558.39 5913.19 1559.57 5913.19 1560.87 5913.19 1562.16 5913.19 1563.61 5913.19 1565.2 5913.19L1565.27 5896.76"/> <path


transform="matrix(.1,0,0,-.1,0,796)" d="M1657.49 5893.1C1661.53 5899.96 1666.33 5905.02 1671.9 5908.28 1677.5 5911.55 1684.1 5913.19 1691.7 5913.19 1701.89 5913.19 1709.75 5909.67 1715.28 5902.63 1720.83 5895.61 1723.6 5885.62 1723.6 5872.64V5812.29H1707.18V5872.12C1707.18 5881.27 1705.53 5888.05 1702.23 5892.47 1698.93 5896.9 1693.88 5899.11 1687.08 5899.11 1678.8 5899.11 1672.25 5896.39 1667.43 5890.97 1662.64 5885.57 1660.24 5878.18 1660.24 5868.82V5812.29H1643.82V5872.12C1643.82 5881.31 1642.17 5888.11 1638.87 5892.51 1635.57 5896.91 1630.47 5899.11 1623.58 5899.11 1615.39 5899.11 1608.89 5896.38 1604.07 5890.93 1599.28 5885.48 1596.88 5878.11 1596.88 5868.82V5812.29H1580.46V5910.84H1596.88V5896.76C1600.55 5902.38 1604.94 5906.53 1610.05 5909.19 1615.18 5911.86 1621.27 5913.19 1628.31 5913.19 1635.42 5913.19 1641.46 5911.46 1646.42 5908.02 1651.41 5904.6 1655.1 5899.62 1657.49 5893.1"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6085.46 4598.23H7906.54V5508.773H6085.46Z" fill="#ffffff" fill-rule="evenodd"/> <path transform="matrix(.1,0,0,-.1,0,796)" stroke-width="2.8346" stroke-linecap="butt" stroke-miterlimit="10" stroke-linejoin="miter" fill="none" stroke="#ffffff" d="M6085.46 4598.23H7906.54V5508.773H6085.46Z"/> <path transform="matrix(.1,0,0,-.1,0,796)" stroke-width="30.3515" stroke-linecap="butt" stroke-dasharray="60.7027" stroke-miterlimit="10" stroke-linejoin="miter" fill="none" stroke="#000000" d="M6085.46 4598.23H7906.54V5508.773H6085.46Z"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6278.6 5212.87V5111.96H6299.72C6317.57 5111.96 6330.63 5116.02 6338.92 5124.14 6347.2 5132.25 6351.35 5145.06 6351.35 5162.56 6351.35 5179.95 6347.2 5192.67 6338.92 5200.73 6330.63 5208.82 6317.57 5212.87 6299.72 5212.87H6278.6M6259.83 5226.95H6297.04C6322.22 5226.95 6340.69 5221.74 6352.45 5211.33 6364.23 5200.94 6370.12 5184.69 6370.12 5162.56 6370.12 5140.3 6364.2 5123.94 6352.38 5113.5 6340.54 5103.09 6322.1 5097.89 6297.04 5097.89H6259.83V5226.95"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6483.78 5152.15V5144.82H6408.69C6409.39 5133.36 6412.75 5124.61 6418.73 5118.6 6424.72 5112.61 6433.06 5109.62 6443.74 5109.62 6449.95 5109.62 6455.96 5110.4 6461.78 5111.96 6467.6 5113.53 6473.37 5115.88 6479.09 5119V5104.93C6473.32 5101.87 6467.4 5099.54 6461.34 5097.92 6455.3 5096.33 6449.18 5095.54 6442.97 5095.54 6427.38 5095.54 6415.02 5100.1 6405.9 5109.21 6396.81 5118.36 6392.26 5130.71 6392.26 5146.29 6392.26 5162.39 6396.63 5175.16 6405.35 5184.6 6414.08 5194.06 6425.85 5198.79 6440.66 5198.79 6453.96 5198.79 6464.47 5194.61 6472.19 5186.25 6479.92 5177.89 6483.78 5166.52 6483.78 5152.15M6467.35 5156.55C6467.23 5165.11 6464.73 5171.94 6459.84 5177.05 6454.97 5182.16 6448.53 5184.71 6440.51 5184.71 6431.42 5184.71 6424.14 5182.24 6418.66 5177.3 6413.21 5172.37 6410.07 5165.41 6409.24 5156.44L6467.35 5156.55"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6581.56 5147.16C6581.56 5158.88 6579.15 5168.05 6574.34 5174.7 6569.55 5181.38 6562.96 5184.71 6554.57 5184.71 6546.19 5184.71 6539.59 5181.38 6534.77 5174.7 6529.98 5168.05 6527.59 5158.88 6527.59 5147.16 6527.59 5135.46 6529.98 5126.27 6534.77 5119.59 6539.59 5112.94 6546.19 5109.62 6554.57 5109.62 6562.96 5109.62 6569.55 5112.94 6574.34 5119.59 6579.15 5126.27 6581.56 5135.46 6581.56 5147.16M6527.59 5182.36C6530.91 5187.91 6535.12 5192.03 6540.2 5194.72 6545.29 5197.43 6551.36 5198.79 6558.43 5198.79 6570.16 5198.79 6579.68 5194.05 6586.99 5184.56 6594.32 5175.08 6597.99 5162.61 6597.99 5147.16 6597.99 5131.71 6594.32 5119.25 6586.99 5109.77 6579.68 5100.28 6570.16 5095.54 6558.43 5095.54 6551.36 5095.54 6545.29 5096.88 6540.2 5099.57 6535.12 5102.29 6530.91 5106.42 6527.59 5111.96V5097.89H6511.16V5233.99H6527.59V5182.36"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6626.56 5136.02V5196.45H6642.99V5136.64C6642.99 5127.64 6644.77 5120.89 6648.34 5116.36 6651.91 5111.87 6657.28 5109.62 6664.44 5109.62 6673.02 5109.62 6679.8 5112.32 6684.79 5117.72 6689.78 5123.15 6692.27 5130.53 6692.27 5139.87V5196.45H6708.7V5097.89H6692.27V5111.96C6688.38 5106.42 6683.86 5102.29 6678.7 5099.57 6673.57 5096.88 6667.61 5095.54 6660.81 5095.54 6649.59 5095.54 6641.07 5098.97 6635.25 5105.84 6629.46 5112.73 6626.56 5122.79 6626.56 5136.02M6667.12 5198.79V5198.79"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6804.61 5148.3C6804.61 5159.86 6802.24 5168.82 6797.5 5175.18 6792.79 5181.53 6786.15 5184.71 6777.59 5184.71 6769.09 5184.71 6762.46 5181.53 6757.72 5175.18 6753 5168.82 6750.64 5159.86 6750.64 5148.3 6750.64 5136.79 6753 5127.86 6757.72 5121.5 6762.46 5115.14 6769.09 5111.96 6777.59 5111.96 6786.15 5111.96 6792.79 5115.14 6797.5 5121.5

6802.24 5127.86 6804.61 5136.79 6804.61 5148.3M6821.04 5110.61C6821.04 5093.64 6817.36 5081.03 6810 5072.77 6802.67 5064.48 6791.44 5060.34 6776.31 5060.34 6770.69 5060.34 6765.39 5060.73 6760.43 5061.51 6755.47 5062.27 6750.64 5063.44 6745.95 5065.03V5081.46C6750.59 5079.06 6755.18 5077.29 6759.7 5076.14 6764.25 5074.99 6768.86 5074.42 6773.56 5074.42 6783.95 5074.42 6791.72 5077.17 6796.88 5082.67 6802.04 5088.17 6804.61 5096.47 6804.61 5107.57V5114.31C6801.34 5108.81 6797.15 5104.69 6792.04 5101.95 6786.93 5099.24 6780.8 5097.89 6773.67 5097.89 6761.84 5097.89 6752.3 5102.48 6745.07 5111.67 6737.83 5120.89 6734.21 5133.1 6734.21 5148.3 6734.21 5163.55 6737.83 5175.78 6745.07 5184.97 6752.3 5194.18 6761.84 5198.79 6773.67 5198.79 6780.8 5198.79 6786.93 5197.42 6792.04 5194.68 6797.15 5191.97 6801.34 5187.86 6804.61 5182.36V5196.45H6821.04V5110.61"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6895.23
5226.95H7007.87V5212.87H6960.94V5097.89H6942.16V5212.87H6895.23V5226.95"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M7053.86 5182.36C7052.11 5183.17 7050.2 5183.76 7048.14 5184.13 7046.09 5184.52 7043.82 5184.71 7041.32 5184.71 7032.52 5184.71 7025.77 5181.77 7021.05 5175.88 7016.33 5170.01 7013.97 5161.57 7013.97 5150.57V5097.89H6997.54V5196.45H7013.97V5182.36C7017.25 5187.91 7021.51 5192.03 7026.77 5194.72 7032.02 5197.43 7038.4 5198.79 7045.91 5198.79 7046.98 5198.79 7048.17 5198.79 7049.46 5198.79 7050.76 5198.79 7052.2 5198.79 7053.79 5198.79L7053.86 5182.36"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M7114.93 5147.16C7102.27 5147.16 7093.49 5145.71 7088.6 5142.8 7083.71 5139.89 7081.27 5134.92 7081.27 5127.88 7081.27 5122.28 7083.1 5117.83 7086.77 5114.53 7090.44 5111.26 7095.42 5109.62 7101.73 5109.62 7110.43 5109.62 7117.41 5112.7 7122.66 5118.86 7127.92 5125.04 7130.55 5133.25 7130.55 5143.5V5147.16H7114.93M7146.98 5153.36V5097.89H7130.55V5111.96C7126.93 5106.34 7122.42 5102.2 7117.02 5099.54 7111.62 5096.87 7105 5095.54 7097.18 5095.54 7087.28 5095.54 7079.41 5098.3 7073.57 5103.82 7067.75 5109.38 7064.84 5116.79 7064.84 5126.08 7064.84 5136.93 7068.5 5145.11 7075.8 5150.61 7083.11 5156.14 7094.03 5158.9 7108.55 5158.9H7130.55V5160.66C7130.55 5168.31 7128.14 5174.22 7123.32 5178.4 7118.54 5182.61 7111.8 5184.71 7103.12 5184.71 7097.6 5184.71 7092.22 5184.13 7086.99 5182.95 7081.76 5181.78 7076.72 5180.02 7071.88 5177.67V5191.75C7077.73 5194.1 7083.39 5195.86 7088.89 5197.03 7094.42 5198.2 7099.79 5198.79 7104.99 5198.79 7119.07 5198.79 7129.58 5195.03 7136.53 5187.5 7143.49 5179.99 7146.98 5168.61 7146.98 5153.36"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M7265.17 5158.24V5097.89H7248.74V5157.72C7248.74 5166.75 7246.95 5173.49 7243.35 5177.96 7239.78 5182.46 7234.42 5184.71 7227.25 5184.71 7218.65 5184.71 7211.87 5182 7206.91 5176.57 7201.94 5171.17 7199.46 5163.79 7199.46 5154.43V5097.89H7183.04V5196.45H7199.46V5182.36C7203.3 5187.86 7207.82 5191.97 7213.03 5194.68 7218.23 5197.42 7224.24 5198.79 7231.03 5198.79 7242.28 5198.79 7250.77 5195.36 7256.52 5188.49 7262.29 5181.64 7265.17 5171.56 7265.17 5158.24"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M7358.74 5191.75V5177.67C7354.34 5180.02 7349.77 5181.78 7345.03 5182.95 7340.29 5184.13 7335.37 5184.71 7330.29 5184.71 7322.56 5184.71 7316.77 5183.5 7312.91 5181.08 7309.04 5178.66 7307.11 5175.03 7307.11 5170.19 7307.11 5166.5 7308.46 5163.6 7311.15 5161.5 7313.86 5159.4 7319.31 5157.39 7327.5 5155.49L7332.74 5154.28C7343.84 5151.83 7351.72 5148.38 7356.39 5143.9 7361.09 5139.43 7363.43 5133.18 7363.43 5125.16 7363.43 5116.05 7359.94 5108.82 7352.95 5103.5 7345.96 5098.19 7336.34 5095.54 7324.09 5095.54 7318.98 5095.54 7313.66 5096.13 7308.14 5097.3 7302.62 5098.47 7296.8 5100.23 7290.69 5102.58V5119C7296.43 5115.88 7302.09 5113.53 7307.66 5111.96 7313.24 5110.4 7318.75 5109.62 7324.2 5109.62 7331.54 5109.62 7337.17 5110.9 7341.1 5113.47 7345.04 5116.06 7347.01 5119.69 7347.01 5124.36 7347.01 5128.71 7345.61 5132.04 7342.83 5134.33 7340.04 5136.65 7333.92 5138.89 7324.46 5141.04L7319.14 5142.32C7309.09 5144.43 7301.83 5147.65 7297.36 5152 7292.91 5156.36 7290.69 5162.32 7290.69 5169.9 7290.69 5179.11 7293.91 5186.23 7300.37 5191.24 7306.82 5196.27 7315.98 5198.79 7327.83 5198.79 7333.7 5198.79 7339.22 5198.2 7344.4 5197.03 7349.59 5195.86 7354.36 5194.1 7358.74 5191.75"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M7408.32
5111.96V5060.34H7391.89V5196.45H7408.32V5182.36C7411.64 5187.91 7415.84 5192.03 7420.93 5194.72 7426.01 5197.43 7432.09 5198.79 7439.15 5198.79 7450.88 5198.79 7460.41 5194.05 7467.71

5184.56 7475.05 5175.08 7478.71 5162.61 7478.71 5147.16 7478.71 5131.71 7475.05 5119.25 7467.71 5109.77 7460.41 5100.28 7450.88 5095.54 7439.15 5095.54 7432.09 5095.54 7426.01 5096.88 7420.93 5099.57 7415.84 5102.29 7411.64 5106.42 7408.32 5111.96M7462.29 5147.16C7462.29 5158.88 7459.88 5168.05 7455.06 5174.7 7450.27 5181.38 7443.68 5184.71 7435.3 5184.71 7426.92 5184.71 7420.32 5181.38 7415.5 5174.7 7410.71 5168.05 7408.32 5158.88 7408.32 5147.16 7408.32 5135.46 7410.71 5126.27 7415.5 5119.59 7420.32 5112.94 7426.92 5109.62 7435.3 5109.62 7443.68 5109.62 7450.27 5112.94 7455.06 5119.59 7459.88 5126.27 7462.29 5135.46 7462.29 5147.16"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M7544.87 5184.71C7536.17 5184.71 7529.29 5181.36 7524.23 5174.66 7519.2 5167.96 7516.68 5158.8 7516.68 5147.16 7516.68 5135.53 7519.18 5126.36 7524.19 5119.66 7529.23 5112.97 7536.12 5109.62 7544.87 5109.62 7553.53 5109.62 7560.37 5112.98 7565.41 5119.7 7570.46 5126.42 7573 5135.58 7573 5147.16 7573 5158.7 7570.46 5167.84 7565.41 5174.59 7560.37 5181.34 7553.53 5184.71 7544.87 5184.71M7544.84 5198.79C7558.72 5198.79 7569.62 5194.22 7577.54 5185.08 7585.46 5175.96 7589.42 5163.32 7589.42 5147.16 7589.42 5131.05 7585.46 5118.42 7577.54 5109.25 7569.62 5100.11 7558.72 5095.54 7544.84 5095.54 7530.9 5095.54 7519.99 5100.11 7512.09 5109.25 7504.2 5118.42 7500.25 5131.05 7500.25 5147.16 7500.25 5163.32 7504.2 5175.96 7512.09 5185.08 7519.99 5194.22 7530.9 5198.79 7544.84 5198.79"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M7674.76 5182.36C7673 5183.17 7671.1 5183.76 7669.04 5184.13 7666.99 5184.52 7664.71 5184.71 7662.22 5184.71 7653.42 5184.71 7646.66 5181.77 7641.95 5175.88 7637.23 5170.01 7634.87 5161.57 7634.87 5150.57V5097.89H7618.44V5196.45H7634.87V5182.36C7638.14 5187.91 7642.41 5192.03 7647.67 5194.72 7652.92 5197.43 7659.3 5198.79 7666.8 5198.79 7667.88 5198.79 7669.07 5198.79 7670.36 5198.79 7671.66 5198.79 7673.1 5198.79 7674.69 5198.79L7674.76 5182.36"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M7709.21 5224.61V5196.45H7742.06V5184.71H7709.21V5130.96C7709.21 5122.89 7710.29 5117.71 7712.43 5115.41 7714.61 5113.11 7719.02 5111.96 7725.67 5111.96H7742.06V5097.89H7725.67C7713.23 5097.89 7704.64 5100.26 7699.89 5105 7695.15 5109.77 7692.78 5118.42 7692.78 5130.96V5184.71H7681.05V5196.45H7692.78V5224.61H7709.21"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6682.18 5000.18H6709.83L6741.99 4913.21 6774.32 5000.18H6801.86V4871.12H6783.09V4984.6L6750.6 4896.93H6733.44L6700.96 4984.6V4871.12H6682.18V5000.18"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6876.23 4957.94C6867.53 4957.94 6860.65 4954.59 6855.59 4947.89 6850.55 4941.2 6848.04 4932.03 6848.04 4920.39 6848.04 4908.76 6850.54 4899.59 6855.55 4892.89 6860.59 4886.2 6867.48 4882.85 6876.23 4882.85 6884.88 4882.85 6891.73 4886.21 6896.77 4892.93 6901.82 4899.66 6904.36 4908.81 6904.36 4920.39 6904.36 4931.93 6901.82 4941.07 6896.77 4947.82 6891.73 4954.57 6884.88 4957.94 6876.23 4957.94M6876.2 4972.02C6890.08 4972.02 6900.98 4967.45 6908.9 4958.31 6916.82 4949.19 6920.78 4936.55 6920.78 4920.39 6920.78 4904.29 6916.82 4891.65 6908.9 4882.48 6900.98 4873.34 6890.08 4868.77 6876.2 4868.77 6862.26 4868.77 6851.35 4873.34 6843.45 4882.48 6835.55 4891.65 6831.61 4904.29 6831.61 4920.39 6831.61 4936.55 6835.55 4949.19 6843.45 4958.31 6851.35 4967.45 6862.26 4972.02 6876.2 4972.02"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M7013.16 4955.59V5007.22H7029.59V4871.12H7013.16V4885.2C7009.84 4879.65 7005.63 4875.52 7000.55 4872.8 6995.46 4870.11 6989.35 4868.77 6982.21 4868.77 6970.55 4868.77 6961.06 4873.51 6953.72 4883 6946.41 4892.48 6942.76 4904.95 6942.76 4920.39 6942.76 4935.84 6946.41 4948.31 6953.72 4957.8 6961.06 4967.28 6970.55 4972.02 6982.21 4972.02 6989.35 4972.02 6995.46 4970.66 7000.55 4967.95 7005.63 4965.26 7009.84 4961.14 7013.16 4955.59M6959.19 4920.39C6959.19 4908.69 6961.58 4899.5 6966.38 4892.82 6971.16 4886.18 6977.75 4882.85 6986.14 4882.85 6994.52 4882.85 7001.12 4886.18 7005.94 4892.82 7010.75 4899.5 7013.16 4908.69 7013.16 4920.39 7013.16 4932.11 7010.75 4941.29 7005.94 4947.93 7001.12 4954.61 6994.52 4957.94 6986.14 4957.94 6977.75 4957.94 6971.16 4954.61 6966.38 4947.93 6961.58 4941.29 6959.19 4932.11 6959.19 4920.39"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M7065.2 4909.25V4969.68H7081.63V4909.87C7081.63 4900.88 7083.41 4894.12 7086.98 4889.6 7090.55 4885.1 7095.92 4882.85 7103.08 4882.85 7111.66 4882.85 7118.44 4885.55 7123.43 4890.95 7128.41 4896.38 7130.91 4903.76 7130.91


4913.1V4969.68H7147.34V4871.12H7130.91V4885.2C7127.02 4879.65 7122.5 4875.52 7117.34 4872.8 7112.21 4870.11 7106.25 4868.77 7099.45 4868.77 7088.23 4868.77 7079.71 4872.2 7073.89 4879.07 7068.1 4885.96 7065.2 4896.02 7065.2 4909.25M7105.75 4972.02V4972.02"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M7179.89 5007.22H7196.32V4871.12H7179.89V5007.22"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M7314.64 4925.38V4918.05H7239.55C7240.25 4906.59 7243.61 4897.85 7249.59 4891.83 7255.58 4885.84 7263.92 4882.85 7274.6 4882.85 7280.81 4882.85 7286.82 4883.63 7292.64 4885.2 7298.46 4886.76 7304.23 4889.11 7309.95 4892.23V4878.16C7304.18 4875.1 7298.26 4872.77 7292.2 4871.15 7286.16 4869.56 7280.04 4868.77 7273.83 4868.77 7258.23 4868.77 7245.88 4873.33 7236.76 4882.45 7227.67 4891.59 7223.12 4903.95 7223.12 4919.52 7223.12 4935.63 7227.48 4948.4 7236.21 4957.83 7244.94 4967.29 7256.71 4972.02 7271.52 4972.02 7284.82 4972.02 7295.33 4967.84 7303.05 4959.48 7310.78 4951.12 7314.64 4939.75 7314.64 4925.38M7298.21 4929.78C7298.09 4938.34 7295.59 4945.17 7290.7 4950.28 7285.83 4955.39 7279.39 4957.94 7271.38 4957.94 7262.28 4957.94 7255 4955.47 7249.52 4950.54 7244.07 4945.6 7240.93 4938.64 7240.1 4929.67L7298.21 4929.78"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M5933.7 4749.99H7754.78O5V5660.5295H5933.7Z" fill="#ffffff" fill-rule="evenodd"/> <path transform="matrix(.1,0,0,-.1,0,796)" stroke-width="2.8346" stroke-linecap="butt" stroke-miterlimit="10" stroke-linejoin="miter" fill="none" stroke="#ffffff" d="M5933.7 4749.99H7754.78O5V5660.5295H5933.7Z"/> <path transform="matrix(.1,0,0,-.1,0,796)" stroke-width="30.3515" stroke-linecap="butt" stroke-dasharray="60.7027" stroke-miterlimit="10" stroke-linejoin="miter" fill="none" stroke="#000000" d="M5933.7 4749.99H7754.78O5V5660.5295H5933.7Z"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6126.84 5364.63V5263.73H6147.96C6165.8 5263.73 6178.87 5267.79 6187.15 5275.9 6195.44 5284.02 6199.58 5296.82 6199.58 5314.32 6199.58 5331.71 6195.44 5344.43 6187.15 5352.5 6178.87 5360.59 6165.8 5364.63 6147.96 5364.63H6126.84M6108.06 5378.71H6145.28C6170.46 5378.71 6188.93 5373.5 6200.68 5363.09 6212.46 5352.7 6218.36 5336.45 6218.36 5314.32 6218.36 5292.06 6212.44 5275.7 6200.61 5265.27 6188.78 5254.85 6170.34 5249.65 6145.28 5249.65H6108.06V5378.71"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6332.02 5303.91V5296.58H6256.92C6257.63 5285.12 6260.98 5276.38 6266.97 5270.36 6272.96 5264.38 6281.29 5261.38 6291.98 5261.38 6298.18 5261.38 6304.2 5262.16 6310.02 5263.73 6315.83 5265.29 6321.6 5267.64 6327.32 5270.77V5256.69C6321.55 5253.63 6315.64 5251.3 6309.57 5249.68 6303.54 5248.09 6297.41 5247.3 6291.21 5247.3 6275.61 5247.3 6263.25 5251.86 6254.14 5260.98 6245.04 5270.12 6240.5 5282.48 6240.5 5298.05 6240.5 5314.16 6244.86 5326.93 6253.59 5336.36 6262.31 5345.82 6274.08 5350.55 6288.89 5350.55 6302.2 5350.55 6312.7 5346.37 6320.43 5338.01 6328.15 5329.65 6332.02 5318.29 6332.02 5303.91M6315.59 5308.31C6315.46 5316.87 6312.96 5323.7 6308.07 5328.81 6303.21 5333.92 6296.77 5336.47 6288.75 5336.47 6279.66 5336.47 6272.37 5334 6266.89 5329.07 6261.45 5324.13 6258.3 5317.18 6257.47 5308.2L6315.59 5308.31"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6429.8 5298.93C6429.8 5310.64 6427.39 5319.81 6422.57 5326.46 6417.78 5333.14 6411.2 5336.47 6402.81 5336.47 6394.43 5336.47 6387.82 5333.14 6383.01 5326.46 6378.22 5319.81 6375.82 5310.64 6375.82 5298.93 6375.82 5287.22 6378.22 5278.03 6383.01 5271.35 6387.82 5264.7 6394.43 5261.38 6402.81 5261.38 6411.2 5261.38 6417.78 5264.7 6422.57 5271.35 6427.39 5278.03 6429.8 5287.22 6429.8 5298.93M6375.82 5334.13C6379.15 5339.68 6383.35 5343.79 6388.44 5346.48 6393.52 5349.2 6399.59 5350.55 6406.66 5350.55 6418.39 5350.55 6427.91 5345.81 6435.22 5336.32 6442.55 5326.84 6446.22 5314.38 6446.22 5298.93 6446.22 5283.48 6442.55 5271.01 6435.22 5261.53 6427.91 5252.04 6418.39 5247.3 6406.66 5247.3 6399.59 5247.3 6393.52 5248.64 6388.44 5251.33 6383.35 5254.05 6379.15 5258.18 6375.82 5263.73V5249.65H6359.39V5385.75H6375.82V5334.13"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6474.8 5287.78V5348.21H6491.23V5288.4C6491.23 5279.41 6493.01 5272.65 6496.58 5268.13 6500.15 5263.63 6505.51 5261.38 6512.68 5261.38 6521.25 5261.38 6528.04 5264.08 6533.02 5269.48 6538.01 5274.91 6540.5 5282.29 6540.5 5291.63V5348.21H6556.93V5249.65H6540.5V5263.73C6536.62 5258.18 6532.09 5254.05 6526.94 5251.33 6521.8 5248.64 6515.84 5247.3 6509.04 5247.3 6497.82 5247.3 6489.3 5250.73 6483.49

5257.6 6477.7 5264.5 6474.8 5274.55 6474.8 5287.78M6515.35 5350.55V5350.55"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6652.85 5300.06C6652.85 5311.63 6650.48 5320.58 6645.74 5326.94 6641.02 5333.29 6634.38 5336.47 6625.83 5336.47 6617.32 5336.47 6610.7 5333.29 6605.95 5326.94 6601.23 5320.58 6598.88 5311.63 6598.88 5300.06 6598.88 5288.55 6601.23 5279.62 6605.95 5273.26 6610.7 5266.91 6617.32 5263.73 6625.83 5263.73 6634.38 5263.73 6641.02 5266.91 6645.74 5273.26 6650.48 5279.62 6652.85 5288.55 6652.85 5300.06M6669.28 5262.37C6669.28 5245.41 6665.6 5232.79 6658.24 5224.53 6650.91 5216.24 6639.68 5212.1 6624.54 5212.1 6618.92 5212.1 6613.63 5212.49 6608.67 5213.27 6603.7 5214.03 6598.88 5215.2 6594.18 5216.79V5233.22C6598.83 5230.82 6603.41 5229.05 6607.93 5227.9 6612.48 5226.75 6617.1 5226.18 6621.79 5226.18 6632.18 5226.18 6639.96 5228.93 6645.11 5234.43 6650.27 5239.93 6652.85 5248.23 6652.85 5259.33V5266.07C6649.57 5260.57 6645.38 5256.45 6640.27 5253.71 6635.16 5251 6629.04 5249.65 6621.9 5249.65 6610.07 5249.65 6600.54 5254.24 6593.3 5263.43 6586.07 5272.65 6582.45 5284.86 6582.45 5300.06 6582.45 5315.32 6586.07 5327.54 6593.3 5336.73 6600.54 5345.95 6610.07 5350.55 6621.9 5350.55 6629.04 5350.55 6635.16 5349.18 6640.27 5346.45 6645.38 5343.73 6649.57 5339.63 6652.85 5334.13V5348.21H6669.28V5262.37"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6743.46 5378.71H6856.11V5364.63H6809.17V5249.65H6790.4V5364.63H6743.46V5378.71"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6902.1 5334.13C6900.34 5334.93 6898.43 5335.52 6896.38 5335.89 6894.33 5336.28 6892.05 5336.47 6889.56 5336.47 6880.76 5336.47 6874 5333.53 6869.29 5327.64 6864.57 5321.77 6862.21 5313.34 6862.21 5302.34V5249.65H6845.78V5348.21H6862.21V5334.13C6865.48 5339.68 6869.75 5343.79 6875 5346.48 6880.26 5349.2 6886.64 5350.55 6894.14 5350.55 6895.22 5350.55 6896.4 5350.55 6897.7 5350.55 6899 5350.55 6900.44 5350.55 6902.03 5350.55L6902.1 5334.13"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6963.16 5298.93C6950.5 5298.93 6941.73 5297.47 6936.84 5294.56 6931.95 5291.65 6929.5 5286.68 6929.5 5279.64 6929.5 5274.04 6931.34 5269.59 6935 5266.29 6938.67 5263.02 6943.66 5261.38 6949.96 5261.38 6958.67 5261.38 6965.64 5264.46 6970.9 5270.62 6976.16 5276.8 6978.79 5285.02 6978.79 5295.26V5298.93H6963.16M6995.21 5305.12V5249.65H6978.79V5263.73C6975.17 5258.11 6970.66 5253.96 6965.25 5251.3 6959.85 5248.63 6953.24 5247.3 6945.42 5247.3 6935.52 5247.3 6927.65 5250.06 6921.8 5255.59 6915.99 5261.14 6913.08 5268.55 6913.08 5277.84 6913.08 5288.7 6916.73 5296.87 6924.04 5302.37 6931.35 5307.9 6942.27 5310.66 6956.79 5310.66H6978.79V5312.42C6978.79 5320.07 6976.38 5325.98 6971.56 5330.16 6966.77 5334.37 6960.04 5336.47 6951.36 5336.47 6945.83 5336.47 6940.46 5335.89 6935.23 5334.71 6929.99 5333.54 6924.96 5331.78 6920.12 5329.43V5343.51C6925.96 5345.86 6931.63 5347.62 6937.13 5348.79 6942.66 5349.96 6948.02 5350.55 6953.23 5350.55 6967.31 5350.55 6977.82 5346.79 6984.76 5339.26 6991.73 5331.75 6995.21 5320.38 6995.21 5305.12"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M7113.4 5310V5249.65H7096.98V5309.48C7096.98 5318.51 7095.18 5325.25 7091.59 5329.73 7088.02 5334.22 7082.65 5336.47 7075.49 5336.47 7066.89 5336.47 7060.1 5333.76 7055.14 5328.33 7050.18 5322.93 7047.7 5315.55 7047.7 5306.19V5249.65H7031.27V5348.21H7047.7V5334.13C7051.54 5339.63 7056.06 5343.73 7061.27 5346.45 7066.47 5349.18 7072.47 5350.55 7079.27 5350.55 7090.51 5350.55 7099.01 5347.12 7104.75 5340.25 7110.52 5333.41 7113.4 5323.32 7113.4 5310"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M7206.98 5343.51V5329.43C7202.58 5331.78 7198 5333.54 7193.26 5334.71 7188.52 5335.89 7183.61 5336.47 7178.52 5336.47 7170.8 5336.47 7165 5335.26 7161.14 5332.84 7157.28 5330.42 7155.35 5326.79 7155.35 5321.95 7155.35 5318.26 7156.7 5315.36 7159.38 5313.26 7162.1 5311.16 7167.55 5309.16 7175.74 5307.25L7180.98 5306.04C7192.08 5303.59 7199.96 5300.14 7204.63 5295.66 7209.32 5291.19 7211.67 5284.95 7211.67 5276.93 7211.67 5267.81 7208.18 5260.59 7201.18 5255.26 7194.19 5249.95 7184.57 5247.3 7172.33 5247.3 7167.22 5247.3 7161.9 5247.89 7156.38 5249.06 7150.85 5250.23 7145.04 5251.99 7138.92 5254.34V5270.77C7144.67 5267.64 7150.33 5265.29 7155.9 5263.73 7161.47 5262.16 7166.98 5261.38 7172.44 5261.38 7179.77 5261.38 7185.4 5262.66 7189.34 5265.23 7193.27 5267.82 7195.24 5271.45 7195.24 5276.12 7195.24 5280.47 7193.85 5283.8 7191.06 5286.09 7188.28 5288.41 7182.15 5290.65

7172.69 5292.8L7167.38 5294.09C7157.33 5296.19 7150.07 5299.41 7145.6 5303.77 7141.15 5308.12 7138.92 5314.08 7138.92 5321.66 7138.92 5330.88 7142.15 5337.99 7148.6 5343 7155.05 5348.04 7164.21 5350.55 7176.07 5350.55 7181.93 5350.55 7187.46 5349.96 7192.64 5348.79 7197.82 5347.62 7202.6 5345.86 7206.98 5343.51"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M7256.55 5263.73V5212.1H7240.13V5348.21H7256.55V5334.13C7259.88 5339.68 7264.08 5343.79 7269.16 5346.48 7274.25 5349.2 7280.32 5350.55 7287.39 5350.55 7299.12 5350.55 7308.64 5345.81 7315.95 5336.32 7323.29 5326.84 7326.95 5314.38 7326.95 5298.93 7326.95 5283.48 7323.29 5271.01 7315.95 5261.53 7308.64 5252.04 7299.12 5247.3 7287.39 5247.3 7280.32 5247.3 7274.25 5248.64 7269.16 5251.33 7264.08 5254.05 7259.88 5258.18 7256.55 5263.73M7310.52 5298.93C7310.52 5310.64 7308.12 5319.81 7303.3 5326.46 7298.51 5333.14 7291.92 5336.47 7283.54 5336.47 7275.15 5336.47 7268.55 5333.14 7263.74 5326.46 7258.95 5319.81 7256.55 5310.64 7256.55 5298.93 7256.55 5287.22 7258.95 5278.03 7263.74 5271.35 7268.55 5264.7 7275.15 5261.38 7283.54 5261.38 7291.92 5261.38 7298.51 5264.7 7303.3 5271.35 7308.12 5278.03 7310.52 5287.22 7310.52 5298.93"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M7393.11 5336.47C7384.41 5336.47 7377.53 5333.13 7372.46 5326.43 7367.43 5319.73 7364.91 5310.56 7364.91 5298.93 7364.91 5287.29 7367.42 5278.13 7372.43 5271.43 7377.46 5264.73 7384.36 5261.38 7393.11 5261.38 7401.76 5261.38 7408.61 5264.74 7413.64 5271.46 7418.7 5278.18 7421.23 5287.34 7421.23 5298.93 7421.23 5310.46 7418.7 5319.61 7413.64 5326.35 7408.61 5333.1 7401.76 5336.47 7393.11 5336.47M7393.07 5350.55C7406.96 5350.55 7417.86 5345.98 7425.78 5336.84 7433.7 5327.72 7437.66 5315.08 7437.66 5298.93 7437.66 5282.82 7433.7 5270.18 7425.78 5261.01 7417.86 5251.87 7406.96 5247.3 7393.07 5247.3 7379.14 5247.3 7368.23 5251.87 7360.33 5261.01 7352.43 5270.18 7348.48 5282.82 7348.48 5298.93 7348.48 5315.08 7352.43 5327.72 7360.33 5336.84 7368.23 5345.98 7379.14 5350.55 7393.07 5350.55"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M7523 5334.13C7521.24 5334.93 7519.33 5335.52 7517.28 5335.89 7515.23 5336.28 7512.95 5336.47 7510.46 5336.47 7501.66 5336.47 7494.9 5333.53 7490.18 5327.64 7485.46 5321.77 7483.11 5313.34 7483.11 5302.34V5249.65H7466.68V5348.21H7483.11V5334.13C7486.38 5339.68 7490.64 5343.79 7495.9 5346.48 7501.16 5349.2 7507.54 5350.55 7515.04 5350.55 7516.12 5350.55 7517.3 5350.55 7518.6 5350.55 7519.89 5350.55 7521.34 5350.55 7522.93 5350.55L7523 5334.13"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M7557.44 5376.37V5348.21H7590.3V5336.47H7557.44V5282.72C7557.44 5274.65 7558.52 5269.47 7560.67 5267.17 7562.84 5264.88 7567.26 5263.73 7573.91 5263.73H7590.3V5249.65H7573.91C7561.46 5249.65 7552.87 5252.02 7548.13 5256.76 7543.39 5261.53 7541.02 5270.18 7541.02 5282.72V5336.47H7529.29V5348.21H7541.02V5376.37H7557.44"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6530.42 5151.95H6558.07L6590.22 5064.97 6622.56 5151.95H6650.1V5022.88H6631.32V5136.36L6598.84 5048.69H6581.68L6549.19 5136.36V5022.88H6530.42V5151.95"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6724.46 5109.7C6715.77 5109.7 6708.88 5106.36 6703.82 5099.66 6698.79 5092.96 6696.27 5083.79 6696.27 5072.16 6696.27 5060.52 6698.78 5051.36 6703.79 5044.66 6708.82 5037.96 6715.71 5034.61 6724.46 5034.61 6733.12 5034.61 6739.96 5037.97 6745 5044.7 6750.06 5051.42 6752.59 5060.57 6752.59 5072.16 6752.59 5083.7 6750.06 5092.84 6745 5099.59 6739.96 5106.33 6733.12 5109.7 6724.46 5109.7M6724.43 5123.79C6738.31 5123.79 6749.21 5119.21 6757.14 5110.07 6765.05 5100.95 6769.02 5088.32 6769.02 5072.16 6769.02 5056.05 6765.05 5043.41 6757.14 5034.25 6749.21 5025.1 6738.31 5020.53 6724.43 5020.53 6710.5 5020.53 6699.58 5025.1 6691.69 5034.25 6683.79 5043.41 6679.84 5056.05 6679.84 5072.16 6679.84 5088.32 6683.79 5100.95 6691.69 5110.07 6699.58 5119.21 6710.5 5123.79 6724.43 5123.79"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6861.39 5107.36V5158.98H6877.82V5022.88H6861.39V5036.96C6858.07 5031.41 6853.87 5027.28 6848.78 5024.57 6843.7 5021.88 6837.59 5020.53 6830.45 5020.53 6818.79 5020.53 6809.29 5025.27 6801.96 5034.76 6794.65 5044.24 6791 5056.71 6791 5072.16 6791 5087.61 6794.65 5100.07 6801.96 5109.56 6809.29 5119.04 6818.79 5123.79 6830.45 5123.79 6837.59 5123.79 6843.7 5122.43 6848.78 5119.71 6853.87 5117.02 6858.07 5112.91 6861.39 5107.36M6807.42 5072.16C6807.42 5060.45 6809.82 5051.26 6814.61 5044.59 6819.4 5037.94 6825.99 5034.61 6834.37 5034.61 6842.76

5034.61 6849.36 5037.94 6854.17 5044.59 6858.99 5051.26 6861.39 5060.45 6861.39 5072.16 6861.39 5083.87 6858.99 5093.05 6854.17 5099.7 6849.36 5106.37 6842.76 5109.7 6834.37 5109.7 6825.99 5109.7 6819.4 5106.37 6814.61 5099.7 6809.82 5093.05 6807.42 5083.87 6807.42 5072.16"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6913.44 5061.01V5121.44H6929.86V5061.63C6929.86 5052.64 6931.65 5045.88 6935.22 5041.36 6938.79 5036.86 6944.15 5034.61 6951.31 5034.61 6959.89 5034.61 6966.68 5037.31 6971.66 5042.71 6976.65 5048.14 6979.14 5055.52 6979.14 5064.86V5121.44H6995.57V5022.88H6979.14V5036.96C6975.26 5031.41 6970.73 5027.28 6965.58 5024.57 6960.45 5021.88 6954.48 5020.53 6947.68 5020.53 6936.46 5020.53 6927.95 5023.96 6922.13 5030.84 6916.34 5037.73 6913.44 5047.79 6913.44 5061.01M6953.99 5123.79V5123.79"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M7028.13 5158.98H7044.56V5022.88H7028.13V5158.98"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M7162.88 5077.14V5069.81H7087.78C7088.49 5058.35 7091.84 5049.61 7097.83 5043.59 7103.82 5037.61 7112.15 5034.61 7122.84 5034.61 7129.04 5034.61 7135.06 5035.39 7140.88 5036.96 7146.69 5038.52 7152.46 5040.87 7158.18 5044V5029.92C7152.41 5026.86 7146.5 5024.53 7140.43 5022.91 7134.4 5021.32 7128.27 5020.53 7122.07 5020.53 7106.47 5020.53 7094.11 5025.09 7085 5034.21 7075.9 5043.35 7071.36 5055.71 7071.36 5071.28 7071.36 5087.39 7075.72 5100.16 7084.45 5109.59 7093.17 5119.05 7104.94 5123.79 7119.75 5123.79 7133.05 5123.79 7143.56 5119.61 7151.29 5111.24 7159.01 5102.88 7162.88 5091.52 7162.88 5077.14M7146.45 5081.54C7146.32 5090.1 7143.82 5096.93 7138.93 5102.04 7134.07 5107.15 7127.63 5109.7 7119.61 5109.7 7110.52 5109.7 7103.23 5107.23 7097.75 5102.3 7092.3 5097.36 7089.16 5090.41 7088.33 5081.43L7146.45 5081.54"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M5781.93 4901.74H7603.01O4V5812.283H5781.93Z" fill="#ffffff" fill-rule="evenodd"/> <path transform="matrix(.1,0,0,-.1,0,796)" stroke-width="2.8346" stroke-linecap="butt" stroke-miterlimit="10" stroke-linejoin="miter" fill="none" stroke="#ffffff" d="M5781.93 4901.74H7603.01O4V5812.283H5781.93Z"/> <path transform="matrix(.1,0,0,-.1,0,796)" stroke-width="30.3515" stroke-linecap="butt" stroke-miterlimit="10" stroke-linejoin="miter" fill="none" stroke="#000000" d="M5781.93 4901.74H7603.01O4V5812.283H5781.93Z"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M5975.1 5516.39V5415.48H5996.22C6014.07 5415.48 6027.13 5419.54 6035.42 5427.66 6043.7 5435.77 6047.85 5448.58 6047.85 5466.08 6047.85 5483.46 6043.7 5496.18 6035.42 5504.25 6027.13 5512.34 6014.07 5516.39 5996.22 5516.39H5975.1M5956.33 5530.47H5993.54C6018.72 5530.47 6037.19 5525.26 6048.95 5514.85 6060.73 5504.46 6066.62 5488.2 6066.62 5466.08 6066.62 5443.81 6060.7 5427.46 6048.88 5417.02 6037.04 5406.61 6018.6 5401.4 5993.54 5401.4H5956.33V5530.47"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6180.28 5455.67V5448.34H6105.19C6105.89 5436.87 6109.24 5428.13 6115.23 5422.12 6121.22 5416.13 6129.56 5413.14 6140.24 5413.14 6146.45 5413.14 6152.46 5413.92 6158.28 5415.48 6164.1 5417.05 6169.87 5419.39 6175.59 5422.52V5408.44C6169.82 5405.39 6163.9 5403.05 6157.84 5401.44 6151.8 5399.85 6145.68 5399.05 6139.47 5399.05 6123.88 5399.05 6111.52 5403.61 6102.4 5412.73 6093.3 5421.88 6088.76 5434.23 6088.76 5449.8 6088.76 5465.91 6093.12 5478.68 6101.85 5488.12 6110.57 5497.58 6122.35 5502.31 6137.16 5502.31 6150.46 5502.31 6160.97 5498.13 6168.69 5489.77 6176.42 5481.41 6180.28 5470.04 6180.28 5455.67M6163.85 5460.07C6163.73 5468.63 6161.22 5475.46 6156.34 5480.57 6151.47 5485.68 6145.03 5488.23 6137.01 5488.23 6127.92 5488.23 6120.63 5485.76 6115.16 5480.82 6109.71 5475.88 6106.57 5468.93 6105.73 5459.96L6163.85 5460.07"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6278.06 5450.68C6278.06 5462.39 6275.65 5471.57 6270.84 5478.22 6266.05 5484.89 6259.46 5488.23 6251.07 5488.23 6242.69 5488.23 6236.09 5484.89 6231.27 5478.22 6226.48 5471.57 6224.09 5462.39 6224.09 5450.68 6224.09 5438.97 6226.48 5429.78 6231.27 5423.11 6236.09 5416.46 6242.69 5413.14 6251.07 5413.14 6259.46 5413.14 6266.05 5416.46 6270.84 5423.11 6275.65 5429.78 6278.06 5438.97 6278.06 5450.68M6224.09 5485.88C6227.41 5491.43 6231.62 5495.55 6236.7 5498.24 6241.79 5500.95 6247.86 5502.31 6254.92 5502.31 6266.66 5502.31 6276.18 5497.57 6283.48 5488.08 6290.82 5478.6 6294.48 5466.13 6294.48 5450.68 6294.48 5435.23 6290.82 5422.77 6283.48 5413.28 6276.18 5403.8 6266.66 5399.05 6254.92 5399.05 6247.86 5399.05 6241.79 5400.4 6236.7 5403.09 6231.62 5405.8 6227.41 5409.93


6224.09 5415.48V5401.4H6207.66V5537.51H6224.09V5485.88"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6323.06 5439.54V5499.96H6339.49V5440.16C6339.49 5431.16 6341.27 5424.41 6344.84 5419.88 6348.41 5415.38 6353.78 5413.14 6360.94 5413.14 6369.52 5413.14 6376.3 5415.84 6381.29 5421.24 6386.27 5426.66 6388.77 5434.05 6388.77 5443.39V5499.96H6405.2V5401.4H6388.77V5415.48C6384.88 5409.93 6380.36 5405.8 6375.2 5403.09 6370.07 5400.4 6364.11 5399.05 6357.31 5399.05 6346.09 5399.05 6337.57 5402.49 6331.75 5409.36 6325.96 5416.25 6323.06 5426.31 6323.06 5439.54M6363.62 5502.31V5502.31"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6501.11 5451.82C6501.11 5463.38 6498.74 5472.34 6494 5478.7 6489.28 5485.05 6482.64 5488.23 6474.09 5488.23 6465.59 5488.23 6458.96 5485.05 6454.22 5478.7 6449.5 5472.34 6447.14 5463.38 6447.14 5451.82 6447.14 5440.3 6449.5 5431.37 6454.22 5425.02 6458.96 5418.66 6465.59 5415.48 6474.09 5415.48 6482.64 5415.48 6489.28 5418.66 6494 5425.02 6498.74 5431.37 6501.11 5440.3 6501.11 5451.82M6517.54 5414.13C6517.54 5397.16 6513.86 5384.55 6506.5 5376.29 6499.17 5368 6487.94 5363.86 6472.81 5363.86 6467.18 5363.86 6461.89 5364.25 6456.93 5365.03 6451.97 5365.79 6447.14 5366.96 6442.45 5368.55V5384.98C6447.09 5382.58 6451.68 5380.81 6456.2 5379.66 6460.75 5378.51 6465.36 5377.94 6470.06 5377.94 6480.45 5377.94 6488.22 5380.69 6493.38 5386.19 6498.54 5391.69 6501.11 5399.98 6501.11 5411.08V5417.83C6497.84 5412.33 6493.64 5408.21 6488.54 5405.47 6483.43 5402.76 6477.3 5401.4 6470.17 5401.4 6458.34 5401.4 6448.8 5406 6441.57 5415.19 6434.33 5424.41 6430.71 5436.61 6430.71 5451.82 6430.71 5467.07 6434.33 5479.29 6441.57 5488.48 6448.8 5497.7 6458.34 5502.31 6470.17 5502.31 6477.3 5502.31 6483.43 5500.94 6488.54 5498.2 6493.64 5495.49 6497.84 5491.38 6501.11 5485.88V5499.96H6517.54V5414.13"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6591.73 5530.47H6704.37V5516.39H6657.43V5401.4H6638.66V5516.39H6591.73V5530.47"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6750.36 5485.88C6748.6 5486.69 6746.7 5487.27 6744.64 5487.64 6742.59 5488.03 6740.32 5488.23 6737.82 5488.23 6729.02 5488.23 6722.27 5485.28 6717.55 5479.39 6712.83 5473.52 6710.47 5465.09 6710.47 5454.09V5401.4H6694.04V5499.96H6710.47V5485.88C6713.75 5491.43 6718.01 5495.55 6723.27 5498.24 6728.52 5500.95 6734.9 5502.31 6742.41 5502.31 6743.48 5502.31 6744.67 5502.31 6745.96 5502.31 6747.26 5502.31 6748.7 5502.31 6750.29 5502.31L6750.36 5485.88"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6811.43 5450.68C6798.77 5450.68 6789.99 5449.23 6785.1 5446.32 6780.21 5443.41 6777.77 5438.44 6777.77 5431.39 6777.77 5425.8 6779.6 5421.35 6783.27 5418.05 6786.93 5414.77 6791.92 5413.14 6798.23 5413.14 6806.93 5413.14 6813.91 5416.21 6819.16 5422.38 6824.42 5428.56 6827.05 5436.77 6827.05 5447.02V5450.68H6811.43M6843.47 5456.88V5401.4H6827.05V5415.48C6823.43 5409.86 6818.92 5405.72 6813.52 5403.05 6808.12 5400.39 6801.5 5399.05 6793.68 5399.05 6783.78 5399.05 6775.91 5401.82 6770.07 5407.34 6764.25 5412.89 6761.34 5420.31 6761.34 5429.6 6761.34 5440.45 6765 5448.63 6772.3 5454.13 6779.61 5459.65 6790.53 5462.41 6805.05 5462.41H6827.05V5464.18C6827.05 5471.83 6824.64 5477.74 6819.82 5481.92 6815.04 5486.13 6808.3 5488.23 6799.62 5488.23 6794.1 5488.23 6788.72 5487.64 6783.49 5486.47 6778.26 5485.3 6773.22 5483.54 6768.38 5481.19V5495.27C6774.22 5497.61 6779.89 5499.38 6785.39 5500.55 6790.92 5501.72 6796.29 5502.31 6801.49 5502.31 6815.57 5502.31 6826.08 5498.54 6833.02 5491.02 6839.99 5483.51 6843.47 5472.13 6843.47 5456.88"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6961.67 5461.75V5401.4H6945.24V5461.24C6945.24 5470.26 6943.45 5477.01 6939.85 5481.48 6936.28 5485.98 6930.92 5488.23 6923.75 5488.23 6915.15 5488.23 6908.37 5485.52 6903.41 5480.09 6898.44 5474.69 6895.96 5467.3 6895.96 5457.94V5401.4H6879.54V5499.96H6895.96V5485.88C6899.8 5491.38 6904.32 5495.49 6909.53 5498.2 6914.73 5500.94 6920.73 5502.31 6927.53 5502.31 6938.78 5502.31 6947.27 5498.88 6953.02 5492 6958.79 5485.16 6961.67 5475.08 6961.67 5461.75"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M7055.24 5495.27V5481.19C7050.84 5483.54 7046.27 5485.3 7041.53 5486.47 7036.79 5487.64 7031.87 5488.23 7026.79 5488.23 7019.06 5488.23 7013.27 5487.02 7009.41 5484.6 7005.54 5482.18 7003.61 5478.55 7003.61 5473.71 7003.61 5470.02 7004.96 5467.12 7007.64 5465.02 7010.36 5462.92 7015.81 5460.91 7024 5459L7029.24 5457.8C7040.34 5455.35 7048.22 5451.89 7052.89 5447.42 7057.59 5442.95 7059.93 5436.7 7059.93 5428.68 7059.93 5419.57 7056.44 5412.34 7049.45

5407.01 7042.45 5401.71 7032.84 5399.05 7020.59 5399.05 7015.48 5399.05 7010.16 5399.64 7004.64 5400.82 6999.12 5401.99 6993.3 5403.75 6987.19 5406.1V5422.52C6992.93 5419.39 6998.59 5417.05 7004.16 5415.48 7009.74 5413.92 7015.25 5413.14 7020.7 5413.14 7028.03 5413.14 7033.67 5414.42 7037.6 5416.98 7041.54 5419.58 7043.51 5423.21 7043.51 5427.88 7043.51 5432.23 7042.11 5435.55 7039.32 5437.85 7036.54 5440.17 7030.42 5442.41 7020.96 5444.56L7015.64 5445.84C7005.59 5447.95 6998.33 5451.17 6993.86 5455.52 6989.41 5459.88 6987.19 5465.84 6987.19 5473.41 6987.19 5482.63 6990.41 5489.74 6996.87 5494.75 7003.32 5499.79 7012.47 5502.31 7024.33 5502.31 7030.2 5502.31 7035.72 5501.72 7040.9 5500.55 7046.09 5499.38 7050.86 5497.61 7055.24 5495.27"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M7104.81 5415.48V5363.86H7088.39V5499.96H7104.81V5485.88C7108.14 5491.43 7112.34 5495.55 7117.43 5498.24 7122.51 5500.95 7128.59 5502.31 7135.65 5502.31 7147.38 5502.31 7156.91 5497.57 7164.21 5488.08 7171.55 5478.6 7175.21 5466.13 7175.21 5450.68 7175.21 5435.23 7171.55 5422.77 7164.21 5413.28 7156.91 5403.8 7147.38 5399.05 7135.65 5399.05 7128.59 5399.05 7122.51 5400.4 7117.43 5403.09 7112.34 5405.8 7108.14 5409.93 7104.81 5415.48M7158.79 5450.68C7158.79 5462.39 7156.38 5471.57 7151.56 5478.22 7146.77 5484.89 7140.18 5488.23 7131.8 5488.23 7123.42 5488.23 7116.82 5484.89 7112 5478.22 7107.21 5471.57 7104.81 5462.39 7104.81 5450.68 7104.81 5438.97 7107.21 5429.78 7112 5423.11 7116.82 5416.46 7123.42 5413.14 7131.8 5413.14 7140.18 5413.14 7146.77 5416.46 7151.56 5423.11 7156.38 5429.78 7158.79 5438.97 7158.79 5450.68"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M7241.37 5488.23C7232.67 5488.23 7225.79 5484.88 7220.73 5478.18 7215.7 5471.48 7213.18 5462.32 7213.18 5450.68 7213.18 5439.05 7215.68 5429.88 7220.69 5423.18 7225.73 5416.48 7232.62 5413.14 7241.37 5413.14 7250.03 5413.14 7256.87 5416.5 7261.91 5423.22 7266.96 5429.94 7269.5 5439.09 7269.5 5450.68 7269.5 5462.22 7266.96 5471.36 7261.91 5478.11 7256.87 5484.86 7250.03 5488.23 7241.37 5488.23M7241.34 5502.31C7255.22 5502.31 7266.12 5497.74 7274.04 5488.59 7281.96 5479.48 7285.92 5466.84 7285.92 5450.68 7285.92 5434.57 7281.96 5421.94 7274.04 5412.77 7266.12 5403.63 7255.22 5399.05 7241.34 5399.05 7227.4 5399.05 7216.49 5403.63 7208.59 5412.77 7200.7 5421.94 7196.75 5434.57 7196.75 5450.68 7196.75 5466.84 7200.7 5479.48 7208.59 5488.59 7216.49 5497.74 7227.4 5502.31 7241.34 5502.31"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M7371.26 5485.88C7369.5 5486.69 7367.59 5487.27 7365.54 5487.64 7363.49 5488.03 7361.21 5488.23 7358.72 5488.23 7349.92 5488.23 7343.16 5485.28 7338.45 5479.39 7333.73 5473.52 7331.37 5465.09 7331.37 5454.09V5401.4H7314.94V5499.96H7331.37V5485.88C7334.64 5491.43 7338.91 5495.55 7344.16 5498.24 7349.42 5500.95 7355.8 5502.31 7363.3 5502.31 7364.38 5502.31 7365.57 5502.31 7366.86 5502.31 7368.16 5502.31 7369.6 5502.31 7371.19 5502.31L7371.26 5485.88"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M7405.71 5528.12V5499.96H7438.56V5488.23H7405.71V5434.48C7405.71 5426.41 7406.78 5421.23 7408.93 5418.93 7411.11 5416.63 7415.52 5415.48 7422.17 5415.48H7438.56V5401.4H7422.17C7409.73 5401.4 7401.14 5403.77 7396.39 5408.52 7391.65 5413.28 7389.28 5421.94 7389.28 5434.48V5488.23H7377.55V5499.96H7389.28V5528.12H7405.71"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6075.73 5303.7H6103.38L6135.54 5216.73 6167.88 5303.7H6195.41V5174.63H6176.64V5288.12L6144.16 5200.45H6127L6094.51 5288.12V5174.63H6075.73V5303.7"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6269.78 5261.46C6261.08 5261.46 6254.2 5258.11 6249.14 5251.41 6244.1 5244.71 6241.59 5235.55 6241.59 5223.91 6241.59 5212.28 6244.09 5203.11 6249.1 5196.41 6254.14 5189.71 6261.03 5186.37 6269.78 5186.37 6278.43 5186.37 6285.28 5189.73 6290.32 5196.45 6295.38 5203.17 6297.91 5212.33 6297.91 5223.91 6297.91 5235.45 6295.38 5244.59 6290.32 5251.34 6285.28 5258.09 6278.43 5261.46 6269.78 5261.46M6269.75 5275.54C6283.63 5275.54 6294.53 5270.97 6302.45 5261.83 6310.37 5252.71 6314.33 5240.07 6314.33 5223.91 6314.33 5207.8 6310.37 5195.17 6302.45 5186 6294.53 5176.86 6283.63 5172.29 6269.75 5172.29 6255.81 5172.29 6244.9 5176.86 6237 5186 6229.11 5195.17 6225.16 5207.8 6225.16 5223.91 6225.16 5240.07 6229.11 5252.71 6237 5261.83 6244.9 5270.97 6255.81 5275.54 6269.75 5275.54"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6406.71 5259.11V5310.74H6423.14V5174.63H6406.71V5188.71C6403.39 5183.16 6399.18 5179.04 6394.1 5176.32 6389.02 5173.63 6382.9 5172.29 6375.77 5172.29 6364.11 5172.29 6354.61 5177.03 6347.28


5186.52 6339.97 5196 6336.31 5208.46 6336.31 5223.91 6336.31 5239.36 6339.97 5251.83 6347.28 5261.31 6354.61 5270.8 6364.11 5275.54 6375.77 5275.54 6382.9 5275.54 6389.02 5274.18 6394.1 5271.47 6399.18 5268.78 6403.39 5264.66 6406.71 5259.11M6352.74 5223.91C6352.74 5212.21 6355.13 5203.02 6359.93 5196.34 6364.71 5189.69 6371.3 5186.37 6379.69 5186.37 6388.07 5186.37 6394.67 5189.69 6399.49 5196.34 6404.3 5203.02 6406.71 5212.21 6406.71 5223.91 6406.71 5235.62 6404.3 5244.8 6399.49 5251.45 6394.67 5258.13 6388.07 5261.46 6379.69 5261.46 6371.3 5261.46 6364.71 5258.13 6359.93 5251.45 6355.13 5244.8 6352.74 5235.62 6352.74 5223.91"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6458.75 5212.77V5273.2H6475.18V5213.39C6475.18 5204.39 6476.96 5197.64 6480.54 5193.11 6484.1 5188.62 6489.47 5186.37 6496.63 5186.37 6505.21 5186.37 6511.99 5189.07 6516.98 5194.47 6521.96 5199.9 6524.46 5207.28 6524.46 5216.62V5273.2H6540.89V5174.63H6524.46V5188.71C6520.57 5183.16 6516.05 5179.04 6510.89 5176.32 6505.76 5173.63 6499.8 5172.29 6493 5172.29 6481.78 5172.29 6473.26 5175.72 6467.45 5182.59 6461.65 5189.48 6458.75 5199.54 6458.75 5212.77M6499.31 5275.54V5275.54"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6573.45 5310.74H6589.87V5174.63H6573.45V5310.74"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6708.19 5228.9V5221.57H6633.1C6633.81 5210.1 6637.16 5201.36 6643.14 5195.35 6649.13 5189.36 6657.47 5186.37 6668.15 5186.37 6674.36 5186.37 6680.38 5187.15 6686.19 5188.71 6692.01 5190.28 6697.78 5192.63 6703.5 5195.75V5181.68C6697.73 5178.62 6691.81 5176.29 6685.75 5174.67 6679.71 5173.08 6673.59 5172.29 6667.38 5172.29 6651.79 5172.29 6639.43 5176.85 6630.31 5185.96 6621.22 5195.11 6616.67 5207.46 6616.67 5223.04 6616.67 5239.14 6621.04 5251.91 6629.76 5261.35 6638.49 5270.81 6650.26 5275.54 6665.07 5275.54 6678.37 5275.54 6688.88 5271.36 6696.61 5263 6704.33 5254.64 6708.19 5243.27 6708.19 5228.9M6691.77 5233.3C6691.64 5241.86 6689.14 5248.69 6684.25 5253.8 6679.38 5258.91 6672.94 5261.46 6664.93 5261.46 6655.83 5261.46 6648.55 5258.99 6643.07 5254.05 6637.62 5249.12 6634.48 5242.16 6633.65 5233.19L6691.77 5233.3"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6832.81 5310.74C6824.87 5297.32 6818.96 5284.05 6815.1 5270.92 6811.27 5257.82 6809.34 5244.52 6809.34 5231.03 6809.34 5217.56 6811.29 5204.21 6815.18 5190.99 6819.06 5177.79 6824.94 5164.52 6832.81 5151.17H6818.59C6810 5164.88 6803.57 5178.35 6799.3 5191.57 6795.05 5204.82 6792.92 5217.97 6792.92 5231.03 6792.92 5244.03 6795.03 5257.12 6799.26 5270.3 6803.49 5283.5 6809.93 5296.98 6818.59 5310.74H6832.81"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6882.49 5289.62V5188.71H6903.61C6921.45 5188.71 6934.52 5192.77 6942.8 5200.89 6951.09 5209 6955.24 5221.81 6955.24 5239.31 6955.24 5256.7 6951.09 5269.42 6942.8 5277.48 6934.52 5285.57 6921.45 5289.62 6903.61 5289.62H6882.49M6863.72 5303.7H6900.93C6926.11 5303.7 6944.58 5298.49 6956.34 5288.08 6968.12 5277.69 6974.01 5261.44 6974.01 5239.31 6974.01 5217.05 6968.09 5200.69 6956.26 5190.25 6944.43 5179.84 6925.99 5174.63 6900.93 5174.63H6863.72V5303.7"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6984.41 5303.7H7097.05V5289.62H7050.12V5174.63H7031.35V5289.62H6984.41V5303.7"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M7113.63 5303.7H7141.28L7173.44 5216.73 7205.78 5303.7H7233.31V5174.63H7214.54V5288.12L7182.05 5200.45H7164.89L7132.41 5288.12V5174.63H7113.63V5303.7"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M7267.75 5310.74H7281.98C7290.56 5296.98 7296.97 5283.5 7301.23 5270.3 7305.5 5257.12 7307.64 5244.03 7307.64 5231.03 7307.64 5217.97 7305.5 5204.82 7301.23 5191.57 7296.97 5178.35 7290.56 5164.88 7281.98 5151.17H7267.75C7275.62 5164.52 7281.5 5177.79 7285.39 5190.99 7289.27 5204.21 7291.22 5217.56 7291.22 5231.03 7291.22 5244.52 7289.27 5257.82 7285.39 5270.92 7281.5 5284.05 7275.62 5297.32 7267.75 5310.74"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M5253.03 2563.93H6642.54V3075.785H5253.03Z" fill="#ffffff" fill-rule="evenodd"/> <path transform="matrix(.1,0,0,-.1,0,796)" stroke-width="2.8346" stroke-linecap="butt" stroke-miterlimit="10" stroke-linejoin="miter" fill="none" stroke="#ffffff" d="M5253.03 2563.93H6642.54V3075.785H5253.03Z"/> <path transform="matrix(.1,0,0,-.1,0,796)" stroke-width="27.7901" stroke-linecap="butt" stroke-miterlimit="10" stroke-linejoin="miter" fill="none" stroke="#000000" d="M5253.03 2563.93H6642.54V3075.785H5253.03Z"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M5407.91 2949.58V2848.68H5429.03C5446.88 2848.68

5459.94 2852.73 5468.23 2860.85 5476.52 2868.97 5480.66 2881.78 5480.66 2899.28 5480.66 2916.66 5476.52 2929.38 5468.23 2937.45 5459.94 2945.54 5446.88 2949.58 5429.03 2949.58H5407.91M5389.14 2963.66H5426.36C5451.53 2963.66 5470 2958.46 5481.76 2948.04 5493.54 2937.66 5499.43 2921.4 5499.43 2899.28 5499.43 2877.01 5493.52 2860.66 5481.68 2850.22 5469.85 2839.8 5451.41 2834.6 5426.36 2834.6H5389.14V2963.66"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M5613.09 2888.86V2881.53H5538C5538.71 2870.07 5542.05 2861.33 5548.04 2855.32 5554.03 2849.32 5562.37 2846.33 5573.05 2846.33 5579.26 2846.33 5585.27 2847.11 5591.09 2848.68 5596.91 2850.24 5602.68 2852.59 5608.39 2855.72V2841.64C5602.63 2838.58 5596.71 2836.25 5590.65 2834.64 5584.61 2833.05 5578.49 2832.25 5572.28 2832.25 5556.68 2832.25 5544.33 2836.81 5535.21 2845.93 5526.12 2855.07 5521.57 2867.43 5521.57 2883 5521.57 2899.11 5525.93 2911.88 5534.66 2921.31 5543.39 2930.77 5555.16 2935.5 5569.97 2935.5 5583.27 2935.5 5593.78 2931.32 5601.5 2922.96 5609.23 2914.61 5613.09 2903.24 5613.09 2888.86M5596.66 2893.27C5596.54 2901.82 5594.04 2908.65 5589.14 2913.76 5584.28 2918.87 5577.84 2921.43 5569.82 2921.43 5560.73 2921.43 5553.45 2918.96 5547.97 2914.02 5542.52 2909.08 5539.38 2902.13 5538.55 2893.16L5596.66 2893.27"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M5710.87 2883.88C5710.87 2895.59 5708.46 2904.77 5703.65 2911.41 5698.86 2918.09 5692.27 2921.43 5683.88 2921.43 5675.5 2921.43 5668.9 2918.09 5664.09 2911.41 5659.29 2904.77 5656.9 2895.59 5656.9 2883.88 5656.9 2872.17 5659.29 2862.98 5664.09 2856.3 5668.9 2849.66 5675.5 2846.33 5683.88 2846.33 5692.27 2846.33 5698.86 2849.66 5703.65 2856.3 5708.46 2862.98 5710.87 2872.17 5710.87 2883.88M5656.9 2919.08C5660.22 2924.63 5664.43 2928.75 5669.51 2931.43 5674.59 2934.15 5680.67 2935.5 5687.73 2935.5 5699.47 2935.5 5708.99 2930.76 5716.3 2921.28 5723.63 2911.79 5727.3 2899.33 5727.3 2883.88 5727.3 2868.43 5723.63 2855.96 5716.3 2846.48 5708.99 2836.99 5699.47 2832.25 5687.73 2832.25 5680.67 2832.25 5674.59 2833.6 5669.51 2836.29 5664.43 2839 5660.22 2843.13 5656.9 2848.68V2834.6H5640.47V2970.7H5656.9V2919.08"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M5755.87 2872.73V2933.16H5772.3V2873.36C5772.3 2864.36 5774.08 2857.6 5777.65 2853.08 5781.22 2848.58 5786.59 2846.33 5793.75 2846.33 5802.33 2846.33 5809.11 2849.03 5814.1 2854.43 5819.09 2859.86 5821.58 2867.24 5821.58 2876.58V2933.16H5838V2834.6H5821.58V2848.68C5817.69 2843.13 5813.17 2839 5808.01 2836.29 5802.88 2833.6 5796.91 2832.25 5790.12 2832.25 5778.9 2832.25 5770.38 2835.69 5764.56 2842.55 5758.77 2849.45 5755.87 2859.51 5755.87 2872.73M5796.43 2935.5V2935.5"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M5933.93 2885.02C5933.93 2896.58 5931.55 2905.54 5926.81 2911.89 5922.09 2918.25 5915.46 2921.43 5906.9 2921.43 5898.39 2921.43 5891.77 2918.25 5887.03 2911.89 5882.31 2905.54 5879.95 2896.58 5879.95 2885.02 5879.95 2873.5 5882.31 2864.57 5887.03 2858.21 5891.77 2851.86 5898.39 2848.68 5906.9 2848.68 5915.46 2848.68 5922.09 2851.86 5926.81 2858.21 5931.55 2864.57 5933.93 2873.5 5933.93 2885.02M5950.35 2847.32C5950.35 2830.36 5946.67 2817.74 5939.31 2809.48 5931.98 2801.2 5920.75 2797.05 5905.62 2797.05 5900 2797.05 5894.7 2797.44 5889.74 2798.23 5884.78 2798.98 5879.95 2800.16 5875.26 2801.75V2818.17C5879.9 2815.78 5884.48 2814 5889.01 2812.86 5893.55 2811.71 5898.18 2811.13 5902.87 2811.13 5913.26 2811.13 5921.03 2813.88 5926.19 2819.38 5931.34 2824.88 5933.93 2833.18 5933.93 2844.28V2851.02C5930.65 2845.52 5926.46 2841.41 5921.35 2838.67 5916.24 2835.95 5910.12 2834.6 5902.98 2834.6 5891.15 2834.6 5881.61 2839.2 5874.38 2848.39 5867.14 2857.6 5863.52 2869.81 5863.52 2885.02 5863.52 2900.27 5867.14 2912.49 5874.38 2921.68 5881.61 2930.89 5891.15 2935.5 5902.98 2935.5 5910.12 2935.5 5916.24 2934.14 5921.35 2931.4 5926.46 2928.68 5930.65 2924.58 5933.93 2919.08V2933.16H5950.35V2847.32"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6043.31 2963.66H6070.96L6103.12 2876.69 6135.46 2963.66H6162.99V2834.6H6144.22V2948.08L6111.73 2860.41H6094.57L6062.09 2948.08V2834.6H6043.31V2963.66"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6237.36 2921.43C6228.66 2921.43 6221.78 2918.07 6216.71 2911.38 6211.68 2904.68 6209.16 2895.51 6209.16 2883.88 6209.16 2872.24 6211.67 2863.07 6216.68 2856.38 6221.71 2849.68 6228.61 2846.33 6237.36 2846.33 6246.01 2846.33 6252.86 2849.69 6257.89 2856.41 6262.95 2863.14 6265.48 2872.29


6265.48 2883.88 6265.48 2895.41 6262.95 2904.56 6257.89 2911.3 6252.86 2918.05 6246.01 2921.43 6237.36 2921.43M6237.32 2935.5C6251.21 2935.5 6262.11 2930.93 6270.03 2921.79 6277.95 2912.67 6281.91 2900.04 6281.91 2883.88 6281.91 2867.77 6277.95 2855.13 6270.03 2845.96 6262.11 2836.82 6251.21 2832.25 6237.32 2832.25 6223.39 2832.25 6212.48 2836.82 6204.58 2845.96 6196.68 2855.13 6192.74 2867.77 6192.74 2883.88 6192.74 2900.04 6196.68 2912.67 6204.58 2921.79 6212.48 2930.93 6223.39 2935.5 6237.32 2935.5"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6374.29 2919.08V2970.7H6390.71V2834.6H6374.29V2848.68C6370.96 2843.13 6366.76 2839 6361.68 2836.29 6356.59 2833.6 6350.48 2832.25 6343.34 2832.25 6331.68 2832.25 6322.19 2836.99 6314.85 2846.48 6307.54 2855.96 6303.89 2868.43 6303.89 2883.88 6303.89 2899.33 6307.54 2911.79 6314.85 2921.28 6322.19 2930.76 6331.68 2935.5 6343.34 2935.5 6350.48 2935.5 6356.59 2934.15 6361.68 2931.43 6366.76 2928.75 6370.96 2924.63 6374.29 2919.08M6320.32 2883.88C6320.32 2872.17 6322.71 2862.98 6327.5 2856.3 6332.29 2849.66 6338.88 2846.33 6347.27 2846.33 6355.65 2846.33 6362.25 2849.66 6367.07 2856.3 6371.88 2862.98 6374.29 2872.17 6374.29 2883.88 6374.29 2895.59 6371.88 2904.77 6367.07 2911.41 6362.25 2918.09 6355.65 2921.43 6347.27 2921.43 6338.88 2921.43 6332.29 2918.09 6327.5 2911.41 6322.71 2904.77 6320.32 2895.59 6320.32 2883.88"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6510.81 2888.86V2881.53H6435.72C6436.43 2870.07 6439.78 2861.33 6445.77 2855.32 6451.75 2849.32 6460.09 2846.33 6470.77 2846.33 6476.98 2846.33 6482.99 2847.11 6488.81 2848.68 6494.63 2850.24 6500.4 2852.59 6506.12 2855.72V2841.64C6500.35 2838.58 6494.43 2836.25 6488.37 2834.64 6482.33 2833.05 6476.21 2832.25 6470 2832.25 6454.41 2832.25 6442.05 2836.81 6432.93 2845.93 6423.84 2855.07 6419.29 2867.43 6419.29 2883 6419.29 2899.11 6423.66 2911.88 6432.38 2921.31 6441.11 2930.77 6452.88 2935.5 6467.69 2935.5 6480.99 2935.5 6491.5 2931.32 6499.23 2922.96 6506.95 2914.61 6510.81 2903.24 6510.81 2888.86M6494.38 2893.27C6494.26 2901.82 6491.76 2908.65 6486.87 2913.76 6482 2918.87 6475.56 2921.43 6467.55 2921.43 6458.45 2921.43 6451.17 2918.96 6445.69 2914.02 6440.24 2909.08 6437.1 2902.13 6436.27 2893.16L6494.38 2893.27"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M622.203 2473.64H2443.283V3080.6629H622.203Z" fill="#ffffff" fill-rule="evenodd"/> <path transform="matrix(.1,0,0,-.1,0,796)" stroke-width="2.8346" stroke-linecap="butt" stroke-miterlimit="10" stroke-linejoin="miter" fill="none" stroke="#ffffff" d="M622.203 2473.64H2443.283V3080.6629H622.203Z"/> <path transform="matrix(.1,0,0,-.1,0,796)" stroke-width="30.3515" stroke-linecap="butt" stroke-dasharray="60.7027" stroke-miterlimit="10" stroke-linejoin="miter" fill="none" stroke="#000000" d="M622.203 2473.64H2443.283V3080.6629H622.203Z"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1122.8 2749.4C1122.8 2761.11 1120.39 2770.29 1115.58 2776.94 1110.79 2783.61 1104.2 2786.95 1095.82 2786.95 1087.43 2786.95 1080.83 2783.61 1076.02 2776.94 1071.23 2770.29 1068.83 2761.11 1068.83 2749.4 1068.83 2737.7 1071.23 2728.5 1076.02 2721.83 1080.83 2715.18 1087.43 2711.86 1095.82 2711.86 1104.2 2711.86 1110.79 2715.18 1115.58 2721.83 1120.39 2728.5 1122.8 2737.7 1122.8 2749.4M1068.83 2784.61C1072.16 2790.15 1076.36 2794.27 1081.45 2796.96 1086.53 2799.67 1092.6 2801.03 1099.67 2801.03 1111.4 2801.03 1120.92 2796.29 1128.23 2786.8 1135.56 2777.32 1139.23 2764.85 1139.23 2749.4 1139.23 2733.96 1135.56 2721.49 1128.23 2712 1120.92 2702.52 1111.4 2697.78 1099.67 2697.78 1092.6 2697.78 1086.53 2699.12 1081.45 2701.81 1076.36 2704.52 1072.16 2708.66 1068.83 2714.2V2700.13H1052.4V2836.23H1068.83V2784.61"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1167.8 2738.26V2798.68H1184.23V2738.88C1184.23 2729.89 1186.02 2723.13 1189.59 2718.61 1193.16 2714.11 1198.52 2711.86 1205.68 2711.86 1214.26 2711.86 1221.05 2714.56 1226.03 2719.96 1231.02 2725.39 1233.51 2732.77 1233.51 2742.11V2798.68H1249.94V2700.13H1233.51V2714.2C1229.63 2708.66 1225.1 2704.52 1219.95 2701.81 1214.81 2699.12 1208.85 2697.78 1202.05 2697.78 1190.83 2697.78 1182.31 2701.21 1176.5 2708.08 1170.7 2714.97 1167.8 2725.04 1167.8 2738.26M1208.36 2801.03V2801.03"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1343.51 2793.99V2779.91C1339.11 2782.26 1334.54 2784.02 1329.8 2785.19 1325.05 2786.36 1320.14 2786.95 1315.06 2786.95 1307.33 2786.95 1301.54 2785.74 1297.68 2783.32 1293.82 2780.9 1291.88 2777.27 1291.88 2772.43 1291.88 2768.74 1293.23 2765.84 1295.92 2763.74 1298.63 2761.64 1304.08 2759.63 1312.27 2757.73L1317.52 2756.52C1328.61 2754.07

1336.5 2750.61 1341.16 2746.14 1345.86 2741.67 1348.2 2735.42 1348.2 2727.41 1348.2 2718.29 1344.71 2711.06 1337.72 2705.73 1330.73 2700.43 1321.11 2697.78 1308.86 2697.78 1303.75 2697.78 1298.44 2698.36 1292.91 2699.54 1287.39 2700.71 1281.57 2702.47 1275.46 2704.82V2721.25C1281.2 2718.12 1286.86 2715.77 1292.43 2714.2 1298.01 2712.64 1303.52 2711.86 1308.97 2711.86 1316.3 2711.86 1321.94 2713.14 1325.88 2715.71 1329.81 2718.3 1331.78 2721.93 1331.78 2726.6 1331.78 2730.95 1330.38 2734.27 1327.6 2736.57 1324.81 2738.89 1318.69 2741.13 1309.23 2743.28L1303.91 2744.56C1293.86 2746.67 1286.61 2749.89 1282.13 2754.24 1277.68 2758.59 1275.46 2764.56 1275.46 2772.14 1275.46 2781.35 1278.68 2788.46 1285.14 2793.48 1291.59 2798.51 1300.75 2801.03 1312.6 2801.03 1318.47 2801.03 1323.99 2800.45 1329.18 2799.27 1334.36 2798.1 1339.14 2796.34 1343.51 2793.99"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1477.05 2749.4C1464.39 2749.4 1455.61 2747.95 1450.73 2745.04 1445.84 2742.13 1443.39 2737.16 1443.39 2730.12 1443.39 2724.52 1445.23 2720.07 1448.89 2716.77 1452.56 2713.5 1457.54 2711.86 1463.85 2711.86 1472.55 2711.86 1479.53 2714.94 1484.79 2721.1 1490.04 2727.28 1492.67 2735.5 1492.67 2745.74V2749.4H1477.05M1509.1 2755.6V2700.13H1492.67V2714.2C1489.05 2708.58 1484.54 2704.44 1479.14 2701.77 1473.74 2699.11 1467.13 2697.78 1459.3 2697.78 1449.41 2697.78 1441.54 2700.54 1435.69 2706.07 1429.88 2711.61 1426.96 2719.03 1426.96 2728.32 1426.96 2739.18 1430.62 2747.35 1437.93 2752.85 1445.24 2758.38 1456.15 2761.14 1470.67 2761.14H1492.67V2762.9C1492.67 2770.55 1490.26 2776.46 1485.45 2780.64 1480.66 2784.85 1473.92 2786.95 1465.25 2786.95 1459.72 2786.95 1454.34 2786.36 1449.11 2785.19 1443.88 2784.02 1438.84 2782.26 1434 2779.91V2793.99C1439.85 2796.34 1445.52 2798.1 1451.02 2799.27 1456.54 2800.45 1461.91 2801.03 1467.11 2801.03 1481.2 2801.03 1491.71 2797.27 1498.65 2789.74 1505.61 2782.23 1509.1 2770.86 1509.1 2755.6"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1615.56 2793.99V2779.91C1610.96 2782.26 1606.36 2784.02 1601.73 2785.19 1597.14 2786.36 1592.49 2786.95 1587.8 2786.95 1577.27 2786.95 1569.09 2783.66 1563.27 2777.09 1557.45 2770.51 1554.54 2761.29 1554.54 2749.4 1554.54 2737.52 1557.45 2728.3 1563.27 2721.72 1569.09 2715.14 1577.27 2711.86 1587.8 2711.86 1592.49 2711.86 1597.14 2712.45 1601.73 2713.62 1606.36 2714.79 1610.96 2716.55 1615.56 2718.9V2704.82C1611.06 2702.47 1606.4 2700.71 1601.59 2699.54 1596.8 2698.36 1591.69 2697.78 1586.26 2697.78 1571.52 2697.78 1559.81 2702.42 1551.13 2711.71 1542.46 2721.02 1538.12 2733.59 1538.12 2749.4 1538.12 2765.44 1542.5 2778.05 1551.28 2787.24 1560.06 2796.43 1572.07 2801.03 1587.32 2801.03 1592.29 2801.03 1597.13 2800.45 1601.84 2799.27 1606.56 2798.1 1611.13 2796.34 1615.56 2793.99"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1715.38 2793.99V2779.91C1710.79 2782.26 1706.18 2784.02 1701.56 2785.19 1696.96 2786.36 1692.32 2786.95 1687.63 2786.95 1677.09 2786.95 1668.91 2783.66 1663.1 2777.09 1657.28 2770.51 1654.37 2761.29 1654.37 2749.4 1654.37 2737.52 1657.28 2728.3 1663.1 2721.72 1668.91 2715.14 1677.09 2711.86 1687.63 2711.86 1692.32 2711.86 1696.96 2712.45 1701.56 2713.62 1706.18 2714.79 1710.79 2716.55 1715.38 2718.9V2704.82C1710.89 2702.47 1706.23 2700.71 1701.41 2699.54 1696.62 2698.36 1691.51 2697.78 1686.09 2697.78 1671.35 2697.78 1659.64 2702.42 1650.96 2711.71 1642.28 2721.02 1637.95 2733.59 1637.95 2749.4 1637.95 2765.44 1642.33 2778.05 1651.11 2787.24 1659.88 2796.43 1671.9 2801.03 1687.15 2801.03 1692.11 2801.03 1696.95 2800.45 1701.67 2799.27 1706.39 2798.1 1710.96 2796.34 1715.38 2793.99"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1829.29 2754.39V2747.06H1754.2C1754.91 2735.59 1758.25 2726.86 1764.24 2720.84 1770.23 2714.85 1778.57 2711.86 1789.25 2711.86 1795.46 2711.86 1801.47 2712.64 1807.29 2714.2 1813.11 2715.77 1818.88 2718.12 1824.59 2721.25V2707.16C1818.83 2704.11 1812.91 2701.77 1806.85 2700.16 1800.81 2698.57 1794.69 2697.78 1788.48 2697.78 1772.88 2697.78 1760.53 2702.34 1751.41 2711.45 1742.32 2720.6 1737.77 2732.95 1737.77 2748.52 1737.77 2764.63 1742.13 2777.41 1750.86 2786.84 1759.59 2796.3 1771.36 2801.03 1786.17 2801.03 1799.46 2801.03 1809.98 2796.85 1817.7 2788.49 1825.43 2780.13 1829.29 2768.77 1829.29 2754.39M1812.86 2758.79C1812.74 2767.35 1810.23 2774.18 1805.34 2779.29 1800.48 2784.39 1794.04 2786.95 1786.02 2786.95 1776.93 2786.95 1769.64 2784.48 1764.17 2779.54 1758.72 2774.61 1755.58 2767.65 1754.75 2758.68L1812.86 2758.79"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1917.68 2793.99V2779.91C1913.28 2782.26 1908.71 2784.02 1903.97 2785.19 1899.23 2786.36 1894.31 2786.95 1889.23 2786.95 1881.5 2786.95 1875.71 2785.74 1871.85 2783.32 1867.99 2780.9 1866.05 2777.27 1866.05 2772.43 1866.05 2768.74 1867.4 2765.84 1870.09 2763.74 1872.8

2761.64 1878.25 2759.63 1886.44 2757.73L1891.69 2756.52C1902.79 2754.07 1910.67 2750.61 1915.34 2746.14 1920.03 2741.67 1922.38 2735.42 1922.38 2727.41 1922.38 2718.29 1918.88 2711.06 1911.89 2705.73 1904.9 2700.43 1895.28 2697.78 1883.03 2697.78 1877.93 2697.78 1872.61 2698.36 1867.08 2699.54 1861.56 2700.71 1855.74 2702.47 1849.63 2704.82V2721.25C1855.38 2718.12 1861.03 2715.77 1866.61 2714.2 1872.18 2712.64 1877.69 2711.86 1883.14 2711.86 1890.48 2711.86 1896.11 2713.14 1900.05 2715.71 1903.98 2718.3 1905.95 2721.93 1905.95 2726.6 1905.95 2730.95 1904.55 2734.27 1901.77 2736.57 1898.98 2738.89 1892.86 2741.13 1883.4 2743.28L1878.08 2744.56C1868.04 2746.67 1860.78 2749.89 1856.3 2754.24 1851.86 2758.59 1849.63 2764.56 1849.63 2772.14 1849.63 2781.35 1852.86 2788.46 1859.31 2793.48 1865.76 2798.51 1874.92 2801.03 1886.77 2801.03 1892.64 2801.03 1898.16 2800.45 1903.35 2799.27 1908.53 2798.1 1913.31 2796.34 1917.68 2793.99"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M2011.84 2793.99V2779.91C2007.45 2782.26 2002.87 2784.02 1998.13 2785.19 1993.39 2786.36 1988.48 2786.95 1983.39 2786.95 1975.67 2786.95 1969.88 2785.74 1966.01 2783.32 1962.15 2780.9 1960.22 2777.27 1960.22 2772.43 1960.22 2768.74 1961.56 2765.84 1964.25 2763.74 1966.96 2761.64 1972.41 2759.63 1980.61 2757.73L1985.85 2756.52C1996.95 2754.07 2004.83 2750.61 2009.5 2746.14 2014.19 2741.67 2016.54 2735.42 2016.54 2727.41 2016.54 2718.29 2013.04 2711.06 2006.05 2705.73 1999.06 2700.43 1989.44 2697.78 1977.2 2697.78 1972.09 2697.78 1966.77 2698.36 1961.25 2699.54 1955.72 2700.71 1949.9 2702.47 1943.79 2704.82V2721.25C1949.54 2718.12 1955.2 2715.77 1960.77 2714.2 1966.34 2712.64 1971.85 2711.86 1977.3 2711.86 1984.64 2711.86 1990.27 2713.14 1994.21 2715.71 1998.14 2718.3 2000.11 2721.93 2000.11 2726.6 2000.11 2730.95 1998.72 2734.27 1995.93 2736.57 1993.14 2738.89 1987.02 2741.13 1977.56 2743.28L1972.25 2744.56C1962.2 2746.67 1954.94 2749.89 1950.46 2754.24 1946.02 2758.59 1943.79 2764.56 1943.79 2772.14 1943.79 2781.35 1947.02 2788.46 1953.47 2793.48 1959.93 2798.51 1969.08 2801.03 1980.93 2801.03 1986.8 2801.03 1992.32 2800.45 1997.51 2799.27 2002.69 2798.1 2007.47 2796.34 2011.84 2793.99"/> <path transform="matrix(.1,0,0,-.1,0,796)" stroke-width="30.3515" stroke-linecap="butt" stroke-miterlimit="10" stroke-linejoin="miter" fill="none" stroke="#000000" d="M2443.28 4598.23 4421.61 4315.62"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M4534.28 4299.52 4373.33 4245.86 4421.61 4315.62 4394.79 4396.09" fill-rule="evenodd"/> <path transform="matrix(.1,0,0,-.1,0,796)" stroke-width="2.8346" stroke-linecap="butt" stroke-miterlimit="10" stroke-linejoin="miter" fill="none" stroke="#000000" d="M4534.28 4299.52 4373.33 4245.86 4421.61 4315.62 4394.79 4396.09Z"/> <path transform="matrix(.1,0,0,-.1,0,796)" stroke-width="30.3515" stroke-linecap="butt" stroke-miterlimit="10" stroke-linejoin="miter" fill="none" stroke="#000000" d="M4534.28 4299.52 4373.33 4245.86 4421.61 4315.62 4394.79 4396.09Z"/> <path transform="matrix(.1,0,0,-.1,0,796)" stroke-width="30.3515" stroke-linecap="butt" stroke-miterlimit="10" stroke-linejoin="miter" fill="none" stroke="#000000" d="M5634.36 5364.39 2894.38 5501.39"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M5748.05 5358.71 5592.69 5290.5 5634.36 5364.39 5600.26 5442.07" fill-rule="evenodd"/> <path transform="matrix(.1,0,0,-.1,0,796)" stroke-width="2.8346" stroke-linecap="butt" stroke-miterlimit="10" stroke-linejoin="miter" fill="none" stroke="#000000" d="M5748.05 5358.71 5592.69 5290.5 5634.36 5364.39 5600.26 5442.07Z"/> <path transform="matrix(.1,0,0,-.1,0,796)" stroke-width="30.3515" stroke-linecap="butt" stroke-miterlimit="10" stroke-linejoin="miter" fill="none" stroke="#000000" d="M5748.05 5358.71 5592.69 5290.5 5634.36 5364.39 5600.26 5442.07Z"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M2780.68 5507.07 2936.05 5575.29 2894.38 5501.39 2928.48 5423.71" fill-rule="evenodd"/> <path transform="matrix(.1,0,0,-.1,0,796)" stroke-width="2.8346" stroke-linecap="butt" stroke-miterlimit="10" stroke-linejoin="miter" fill="none" stroke="#000000" d="M2780.68 5507.07 2936.05 5575.29 2894.38 5501.39 2928.48 5423.71Z"/> <path transform="matrix(.1,0,0,-.1,0,796)" stroke-width="30.3515" stroke-linecap="butt" stroke-miterlimit="10" stroke-linejoin="miter" fill="none" stroke="#000000" d="M2780.68 5507.07 2936.05 5575.29 2894.38 5501.39 2928.48 5423.71Z"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M5191.32 3513.23H5210.1V3461.61H5271.11V3513.23H5289.88V3384.17H5271.11V3447.53H5210.1V3384.17H5191.32V3513.23"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M5371.01 3433.45C5358.35 3433.45 5349.57 3432 5344.69 3429.09 5339.8 3426.18 5337.35 3421.2 5337.35 3414.16 5337.35 3408.57


5339.19 3404.12 5342.85 3400.82 5346.52 3397.54 5351.51 3395.9 5357.81 3395.9 5366.52 3395.9 5373.49 3398.98 5378.75 3405.14 5384 3411.33 5386.63 3419.54 5386.63 3429.79V3433.45H5371.01M5403.06 3439.65V3384.17H5386.63V3398.25C5383.02 3392.63 5378.5 3388.48 5373.1 3385.82 5367.7 3383.16 5361.09 3381.82 5353.27 3381.82 5343.37 3381.82 5335.5 3384.59 5329.65 3390.11 5323.84 3395.66 5320.93 3403.08 5320.93 3412.37 5320.93 3423.22 5324.58 3431.4 5331.89 3436.9 5339.2 3442.42 5350.11 3445.18 5364.63 3445.18H5386.63V3446.94C5386.63 3454.59 5384.23 3460.51 5379.41 3464.69 5374.62 3468.89 5367.88 3471 5359.21 3471 5353.68 3471 5348.3 3470.41 5343.07 3469.24 5337.84 3468.06 5332.8 3466.3 5327.96 3463.96V3478.04C5333.81 3480.38 5339.48 3482.14 5344.98 3483.32 5350.5 3484.49 5355.87 3485.08 5361.07 3485.08 5375.16 3485.08 5385.67 3481.31 5392.61 3473.78 5399.57 3466.28 5403.06 3454.9 5403.06 3439.65"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M5495.44 3468.65C5493.68 3469.46 5491.77 3470.04 5489.72 3470.41 5487.66 3470.8 5485.39 3471 5482.9 3471 5474.1 3471 5467.34 3468.05 5462.62 3462.16 5457.91 3456.29 5455.55 3447.86 5455.55 3436.86V3384.17H5439.12V3482.73H5455.55V3468.65C5458.82 3474.2 5463.09 3478.32 5468.34 3481.01 5473.6 3483.72 5479.98 3485.08 5487.48 3485.08 5488.56 3485.08 5489.74 3485.08 5491.04 3485.08 5492.34 3485.08 5493.78 3485.08 5495.37 3485.08L5495.44 3468.65"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M5573.98 3468.65V3520.28H5590.41V3384.17H5573.98V3398.25C5570.66 3392.7 5566.46 3388.57 5561.37 3385.86 5556.29 3383.17 5550.18 3381.82 5543.04 3381.82 5531.38 3381.82 5521.88 3386.57 5514.55 3396.05 5507.24 3405.54 5503.59 3418 5503.59 3433.45 5503.59 3448.9 5507.24 3461.37 5514.55 3470.85 5521.88 3480.34 5531.38 3485.08 5543.04 3485.08 5550.18 3485.08 5556.29 3483.72 5561.37 3481.01 5566.46 3478.32 5570.66 3474.2 5573.98 3468.65M5520.01 3433.45C5520.01 3421.74 5522.41 3412.55 5527.2 3405.88 5531.99 3399.23 5538.58 3395.9 5546.96 3395.9 5555.35 3395.9 5561.95 3399.23 5566.76 3405.88 5571.58 3412.55 5573.98 3421.74 5573.98 3433.45 5573.98 3445.16 5571.58 3454.34 5566.76 3460.99 5561.95 3467.66 5555.35 3471 5546.96 3471 5538.58 3471 5531.99 3467.66 5527.2 3460.99 5522.41 3454.34 5520.01 3445.16 5520.01 3433.45"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M5616.64 3482.73H5632.74L5652.83 3405.8 5672.85 3482.73H5691.84L5711.94 3405.8 5731.96 3482.73H5748.05L5722.42 3384.17H5703.47L5682.38 3464.95 5661.23 3384.17H5642.27L5616.64 3482.73"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M5817.04 3433.45C5804.38 3433.45 5795.61 3432 5790.71 3429.09 5785.83 3426.18 5783.38 3421.2 5783.38 3414.16 5783.38 3408.57 5785.21 3404.12 5788.88 3400.82 5792.55 3397.54 5797.54 3395.9 5803.84 3395.9 5812.54 3395.9 5819.52 3398.98 5824.78 3405.14 5830.04 3411.33 5832.66 3419.54 5832.66 3429.79V3433.45H5817.04M5849.09 3439.65V3384.17H5832.66V3398.25C5829.04 3392.63 5824.54 3388.48 5819.13 3385.82 5813.73 3383.16 5807.12 3381.82 5799.3 3381.82 5789.39 3381.82 5781.52 3384.59 5775.68 3390.11 5769.86 3395.66 5766.96 3403.08 5766.96 3412.37 5766.96 3423.22 5770.61 3431.4 5777.92 3436.9 5785.23 3442.42 5796.14 3445.18 5810.66 3445.18H5832.66V3446.94C5832.66 3454.59 5830.25 3460.51 5825.44 3464.69 5820.65 3468.89 5813.91 3471 5805.23 3471 5799.71 3471 5794.33 3470.41 5789.1 3469.24 5783.87 3468.06 5778.84 3466.3 5774 3463.96V3478.04C5779.84 3480.38 5785.51 3482.14 5791.01 3483.32 5796.54 3484.49 5801.9 3485.08 5807.11 3485.08 5821.19 3485.08 5831.7 3481.31 5838.64 3473.78 5845.61 3466.28 5849.09 3454.9 5849.09 3439.65"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M5941.47 3468.65C5939.71 3469.46 5937.8 3470.04 5935.75 3470.41 5933.7 3470.8 5931.42 3471 5928.93 3471 5920.13 3471 5913.37 3468.05 5908.65 3462.16 5903.93 3456.29 5901.57 3447.86 5901.57 3436.86V3384.17H5885.15V3482.73H5901.57V3468.65C5904.85 3474.2 5909.12 3478.32 5914.37 3481.01 5919.63 3483.72 5926.01 3485.08 5933.51 3485.08 5934.59 3485.08 5935.77 3485.08 5937.07 3485.08 5938.36 3485.08 5939.8 3485.08 5941.39 3485.08L5941.47 3468.65"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6039.72 3438.44V3431.1H5964.63C5965.33 3419.64 5968.68 3410.9 5974.67 3404.89 5980.66 3398.9 5989 3395.9 5999.68 3395.9 6005.89 3395.9 6011.9 3396.69 6017.72 3398.25 6023.54 3399.82 6029.3 3402.16 6035.02 3405.29V3391.21C6029.25 3388.16 6023.34 3385.82 6017.28 3384.21 6011.24 3382.62 6005.12 3381.82 5998.91 3381.82 5983.31 3381.82 5970.96 3386.38 5961.84 3395.5 5952.75 3404.64 5948.2

3417 5948.2 3432.57 5948.2 3448.68 5952.56 3461.45 5961.29 3470.89 5970.02 3480.35 5981.79 3485.08 5996.6 3485.08 6009.89 3485.08 6020.41 3480.89 6028.13 3472.54 6035.86 3464.18 6039.72 3452.81 6039.72 3438.44M6023.29 3442.84C6023.17 3451.39 6020.66 3458.22 6015.77 3463.33 6010.91 3468.44 6004.47 3471 5996.45 3471 5987.36 3471 5980.07 3468.53 5974.6 3463.59 5969.14 3458.65 5966 3451.7 5965.18 3442.73L6023.29 3442.84"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6105.67 3513.23H6218.31V3499.16H6171.38V3384.17H6152.61V3499.16H6105.67V3513.23"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6317.02 3444.52V3384.17H6300.59V3444.01C6300.59 3453.03 6298.8 3459.78 6295.21 3464.25 6291.64 3468.75 6286.27 3471 6279.11 3471 6270.5 3471 6263.72 3468.28 6258.76 3462.86 6253.8 3457.45 6251.32 3450.07 6251.32 3440.71V3384.17H6234.89V3520.28H6251.32V3468.65C6255.15 3474.15 6259.68 3478.26 6264.88 3480.97 6270.09 3483.71 6276.09 3485.08 6282.89 3485.08 6294.13 3485.08 6302.63 3481.64 6308.37 3474.77 6314.14 3467.93 6317.02 3457.84 6317.02 3444.52"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6405.9 3468.65C6404.14 3469.46 6402.23 3470.04 6400.18 3470.41 6398.13 3470.8 6395.86 3471 6393.36 3471 6384.56 3471 6377.8 3468.05 6373.09 3462.16 6368.37 3456.29 6366.01 3447.86 6366.01 3436.86V3384.17H6349.58V3482.73H6366.01V3468.65C6369.28 3474.2 6373.55 3478.32 6378.8 3481.01 6384.06 3483.72 6390.44 3485.08 6397.95 3485.08 6399.02 3485.08 6400.2 3485.08 6401.5 3485.08 6402.8 3485.08 6404.24 3485.08 6405.83 3485.08L6405.9 3468.65"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6504.15 3438.44V3431.1H6429.05C6429.77 3419.64 6433.11 3410.9 6439.1 3404.89 6445.09 3398.9 6453.43 3395.9 6464.11 3395.9 6470.32 3395.9 6476.33 3396.69 6482.15 3398.25 6487.97 3399.82 6493.73 3402.16 6499.46 3405.29V3391.21C6493.69 3388.16 6487.77 3385.82 6481.71 3384.21 6475.67 3382.62 6469.55 3381.82 6463.34 3381.82 6447.74 3381.82 6435.39 3386.38 6426.27 3395.5 6417.18 3404.64 6412.63 3417 6412.63 3432.57 6412.63 3448.68 6416.99 3461.45 6425.72 3470.89 6434.45 3480.35 6446.21 3485.08 6461.03 3485.08 6474.33 3485.08 6484.84 3480.89 6492.56 3472.54 6500.29 3464.18 6504.15 3452.81 6504.15 3438.44M6487.72 3442.84C6487.6 3451.39 6485.09 3458.22 6480.21 3463.33 6475.34 3468.44 6468.9 3471 6460.88 3471 6451.79 3471 6444.5 3468.53 6439.03 3463.59 6433.58 3458.65 6430.44 3451.7 6429.61 3442.73L6487.72 3442.84"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6574.58 3433.45C6561.91 3433.45 6553.14 3432 6548.25 3429.09 6543.36 3426.18 6540.92 3421.2 6540.92 3414.16 6540.92 3408.57 6542.75 3404.12 6546.42 3400.82 6550.08 3397.54 6555.07 3395.9 6561.38 3395.9 6570.08 3395.9 6577.06 3398.98 6582.31 3405.14 6587.57 3411.33 6590.2 3419.54 6590.2 3429.79V3433.45H6574.58M6606.63 3439.65V3384.17H6590.2V3398.25C6586.58 3392.63 6582.07 3388.48 6576.67 3385.82 6571.27 3383.16 6564.65 3381.82 6556.83 3381.82 6546.93 3381.82 6539.06 3384.59 6533.22 3390.11 6527.4 3395.66 6524.49 3403.08 6524.49 3412.37 6524.49 3423.22 6528.14 3431.4 6535.45 3436.9 6542.76 3442.42 6553.68 3445.18 6568.2 3445.18H6590.2V3446.94C6590.2 3454.59 6587.79 3460.51 6582.97 3464.69 6578.18 3468.89 6571.45 3471 6562.77 3471 6557.25 3471 6551.87 3470.41 6546.64 3469.24 6541.41 3468.06 6536.37 3466.3 6531.53 3463.96V3478.04C6537.37 3480.38 6543.04 3482.14 6548.54 3483.32 6554.07 3484.49 6559.43 3485.08 6564.64 3485.08 6578.72 3485.08 6589.23 3481.31 6596.17 3473.78 6603.14 3466.28 6606.63 3454.9 6606.63 3439.65"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M6706.04 3468.65V3520.28H6722.47V3384.17H6706.04V3398.25C6702.72 3392.7 6698.52 3388.57 6693.43 3385.86 6688.35 3383.17 6682.23 3381.82 6675.1 3381.82 6663.44 3381.82 6653.94 3386.57 6646.61 3396.05 6639.3 3405.54 6635.64 3418 6635.64 3433.45 6635.64 3448.9 6639.3 3461.37 6646.61 3470.85 6653.94 3480.34 6663.44 3485.08 6675.1 3485.08 6682.23 3485.08 6688.35 3483.72 6693.43 3481.01 6698.52 3478.32 6702.72 3474.2 6706.04 3468.65M6652.07 3433.45C6652.07 3421.74 6654.46 3412.55 6659.26 3405.88 6664.05 3399.23 6670.64 3395.9 6679.02 3395.9 6687.41 3395.9 6694 3399.23 6698.82 3405.88 6703.64 3412.55 6706.04 3421.74 6706.04 3433.45 6706.04 3445.16 6703.64 3454.34 6698.82 3460.99 6694 3467.66 6687.41 3471 6679.02 3471 6670.64 3471 6664.05 3467.66 6659.26 3460.99 6654.46 3454.34 6652.07 3445.16 6652.07 3433.45"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M562.039 810.27H2383.119V1417.297H562.039Z" fill="#ffffff"

fill-rule="evenodd"/> <path transform="matrix(.1,0,0,-.1,0,796)" stroke-width="2.8346" stroke-linecap="butt" stroke-miterlimit="10" stroke-linejoin="miter" fill="none" stroke="#ffffff" d="M562.039 810.27H2383.119V1417.297H562.039Z"/> <path transform="matrix(.1,0,0,-.1,0,796)" stroke-width="30.3515" stroke-linecap="butt" stroke-dasharray="60.7027" stroke-miterlimit="10" stroke-linejoin="miter" fill="none" stroke="#000000" d="M562.039 810.27H2383.119V1417.297H562.039Z"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1064.95 1248.82V1199.54H1086.66C1094.7 1199.54 1100.91 1201.68 1105.28 1205.96 1109.68 1210.24 1111.88 1216.32 1111.88 1224.22 1111.88 1232.07 1109.68 1238.13 1105.28 1242.41 1100.91 1246.68 1094.7 1248.82 1086.66 1248.82H1064.95M1046.18 1262.9H1086.66C1101.15 1262.9 1112.1 1259.62 1119.51 1253.04 1126.94 1246.49 1130.66 1236.88 1130.66 1224.22 1130.66 1211.46 1126.94 1201.8 1119.51 1195.25 1112.1 1188.73 1101.15 1185.46 1086.66 1185.46H1064.95V1133.84H1046.18V1262.9"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1208.7 1218.32C1206.93 1219.13 1205.03 1219.71 1202.97 1220.08 1200.92 1220.47 1198.65 1220.66 1196.16 1220.66 1187.36 1220.66 1180.59 1217.72 1175.88 1211.83 1171.16 1205.96 1168.8 1197.53 1168.8 1186.53V1133.84H1152.38V1232.4H1168.8V1218.32C1172.08 1223.87 1176.34 1227.98 1181.6 1230.67 1186.85 1233.39 1193.23 1234.74 1200.74 1234.74 1201.81 1234.74 1203 1234.74 1204.29 1234.74 1205.59 1234.74 1207.03 1234.74 1208.62 1234.74L1208.7 1218.32"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1260.05 1220.66C1251.34 1220.66 1244.46 1217.32 1239.4 1210.62 1234.37 1203.92 1231.85 1194.75 1231.85 1183.12 1231.85 1171.48 1234.36 1162.32 1239.37 1155.62 1244.4 1148.92 1251.3 1145.57 1260.05 1145.57 1268.7 1145.57 1275.54 1148.93 1280.58 1155.66 1285.64 1162.38 1288.17 1171.53 1288.17 1183.12 1288.17 1194.66 1285.64 1203.8 1280.58 1210.54 1275.54 1217.29 1268.7 1220.66 1260.05 1220.66M1260.01 1234.74C1273.89 1234.74 1284.8 1230.17 1292.71 1221.03 1300.64 1211.91 1304.6 1199.27 1304.6 1183.12 1304.6 1167.01 1300.64 1154.37 1292.71 1145.2 1284.8 1136.06 1273.89 1131.49 1260.01 1131.49 1246.08 1131.49 1235.16 1136.06 1227.27 1145.2 1219.37 1154.37 1215.42 1167.01 1215.42 1183.12 1215.42 1199.27 1219.37 1211.91 1227.27 1221.03 1235.16 1230.17 1246.08 1234.74 1260.01 1234.74"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1396.98 1184.25C1396.98 1195.82 1394.61 1204.77 1389.86 1211.13 1385.14 1217.48 1378.51 1220.66 1369.95 1220.66 1361.45 1220.66 1354.82 1217.48 1350.08 1211.13 1345.36 1204.77 1343 1195.82 1343 1184.25 1343 1172.74 1345.36 1163.8 1350.08 1157.45 1354.82 1151.09 1361.45 1147.92 1369.95 1147.92 1378.51 1147.92 1385.14 1151.09 1389.86 1157.45 1394.61 1163.8 1396.98 1172.74 1396.98 1184.25M1413.4 1146.56C1413.4 1129.6 1409.72 1116.98 1402.37 1108.72 1395.03 1100.43 1383.8 1096.29 1368.67 1096.29 1363.05 1096.29 1357.75 1096.68 1352.79 1097.46 1347.83 1098.22 1343 1099.39 1338.31 1100.98V1117.41C1342.95 1115.02 1347.54 1113.24 1352.06 1112.09 1356.61 1110.95 1361.23 1110.37 1365.92 1110.37 1376.31 1110.37 1384.08 1113.12 1389.24 1118.62 1394.4 1124.12 1396.98 1132.42 1396.98 1143.52V1150.27C1393.7 1144.77 1389.51 1140.64 1384.4 1137.91 1379.29 1135.2 1373.17 1133.84 1366.03 1133.84 1354.2 1133.84 1344.66 1138.43 1337.43 1147.63 1330.2 1156.84 1326.58 1169.05 1326.58 1184.25 1326.58 1199.51 1330.2 1211.73 1337.43 1220.92 1344.66 1230.14 1354.2 1234.74 1366.03 1234.74 1373.17 1234.74 1379.29 1233.38 1384.4 1230.64 1389.51 1227.92 1393.7 1223.82 1396.98 1218.32V1232.4H1413.4V1146.56"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1505.34 1218.32C1503.58 1219.13 1501.67 1219.71 1499.62 1220.08 1497.57 1220.47 1495.29 1220.66 1492.8 1220.66 1484 1220.66 1477.24 1217.72 1472.52 1211.83 1467.8 1205.96 1465.45 1197.53 1465.45 1186.53V1133.84H1449.02V1232.4H1465.45V1218.32C1468.72 1223.87 1472.98 1227.98 1478.24 1230.67 1483.5 1233.39 1489.88 1234.74 1497.38 1234.74 1498.46 1234.74 1499.64 1234.74 1500.94 1234.74 1502.23 1234.74 1503.68 1234.74 1505.27 1234.74L1505.34 1218.32"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1566.4 1183.12C1553.74 1183.12 1544.96 1181.66 1540.07 1178.75 1535.19 1175.84 1532.74 1170.87 1532.74 1163.83 1532.74 1158.23 1534.57 1153.79 1538.24 1150.48 1541.91 1147.21 1546.89 1145.57 1553.2 1145.57 1561.91 1145.57 1568.88 1148.65 1574.14 1154.81 1579.39 1161 1582.02 1169.21 1582.02 1179.45V1183.12H1566.4M1598.45 1189.31V1133.84H1582.02V1147.92C1578.41 1142.3 1573.89 1138.15 1568.49 1135.49 1563.09 1132.82 1556.48 1131.49 1548.66 1131.49 1538.76 1131.49 1530.89 1134.25 1525.04 1139.78 1519.23 1145.33 1516.32 1152.75 1516.32 1162.04 1516.32 1172.89 1519.97 1181.06 1527.28 1186.56 1534.59 1192.09 1545.5 1194.85 1560.02 1194.85H1582.02V1196.61C1582.02 1204.26 1579.61 1210.18 1574.8 1214.36 1570.01


1218.56 1563.27 1220.66 1554.6 1220.66 1549.07 1220.66 1543.7 1220.08 1538.46 1218.9 1533.23 1217.73 1528.2 1215.97 1523.36 1213.63V1227.7C1529.2 1230.05 1534.87 1231.81 1540.37 1232.98 1545.89 1234.16 1551.26 1234.74 1556.46 1234.74 1570.55 1234.74 1581.06 1230.98 1588 1223.45 1594.96 1215.95 1598.45 1204.57 1598.45 1189.31"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1711.55 1214.65C1715.58 1221.52 1720.38 1226.58 1725.95 1229.83 1731.55 1233.11 1738.15 1234.74 1745.75 1234.74 1755.95 1234.74 1763.81 1231.22 1769.33 1224.18 1774.88 1217.17 1777.66 1207.17 1777.66 1194.19V1133.84H1761.23V1193.68C1761.23 1202.82 1759.58 1209.6 1756.28 1214.03 1752.98 1218.45 1747.93 1220.66 1741.13 1220.66 1732.85 1220.66 1726.3 1217.95 1721.48 1212.52 1716.69 1207.12 1714.3 1199.74 1714.3 1190.38V1133.84H1697.87V1193.68C1697.87 1202.87 1696.22 1209.66 1692.92 1214.06 1689.62 1218.46 1684.52 1220.66 1677.63 1220.66 1669.44 1220.66 1662.94 1217.94 1658.12 1212.49 1653.33 1207.04 1650.94 1199.67 1650.94 1190.38V1133.84H1634.51V1232.4H1650.94V1218.32C1654.6 1223.94 1658.99 1228.08 1664.1 1230.75 1669.23 1233.41 1675.32 1234.74 1682.36 1234.74 1689.47 1234.74 1695.51 1233.02 1700.47 1229.57 1705.46 1226.15 1709.15 1221.18 1711.55 1214.65"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M742.16 970.43V921.148H769.145C778.313 921.148 785.098 923.18 789.496 927.238 793.922 931.293 796.133 937.492 796.133 945.828 796.133 954.234 793.922 960.434 789.496 964.418 785.098 968.426 778.313 970.43 769.145 970.43H742.16M742.16 1022.05V984.508H767.055C775.27 984.508 781.379 986.051 785.391 989.129 789.422 992.234 791.438 996.953 791.438 1003.28 791.438 1009.57 789.422 1014.26 785.391 1017.36 781.379 1020.49 775.27 1022.05 767.055 1022.05H742.16M723.387 1036.14H768.301C781.723 1036.14 792.063 1033.45 799.324 1028.07 806.582 1022.69 810.211 1015.04 810.211 1005.12 810.211 997.441 808.281 991.328 804.418 986.781 800.559 982.262 794.875 979.438 787.367 978.313 796.07 976.43 802.828 972.496 807.645 966.508 812.484 960.52 814.906 953.039 814.906 944.066 814.906 932.262 811.008 923.141 803.207 916.715 795.41 910.285 784.324 907.07 769.953 907.07H723.387V1036.14"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M847.992 945.203V1005.63H864.418V945.828C864.418 936.832 866.203 930.074 869.773 925.551 873.34 921.051 878.707 918.805 885.867 918.805 894.449 918.805 901.23 921.504 906.219 926.906 911.203 932.332 913.699 939.715 913.699 949.055V1005.63H930.125V907.07H913.699V921.148C909.813 915.602 905.289 911.469 900.133 908.758 895 906.066 889.035 904.723 882.238 904.723 871.02 904.723 862.5 908.156 856.684 915.027 850.891 921.922 847.992 931.98 847.992 945.203M888.547 1007.98V1007.98"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1072.98 1043.18V1029.1H1057.72C1052 1029.1 1048.03 1027.98 1045.81 1025.76 1043.58 1023.56 1042.47 1019.59 1042.47 1013.84V1005.63H1068.28V993.895H1042.47V907.07H1026.04V993.895H981.457V907.07H965.031V993.895H948.605V1005.63H965.031V1012.34C965.031 1023.07 967.563 1030.89 972.621 1035.8 977.707 1040.72 985.746 1043.18 996.746 1043.18H1011.96V1029.1H996.711C990.992 1029.1 987.008 1028 984.758 1025.8 982.559 1023.57 981.457 1019.59 981.457 1013.84V1005.63H1026.04V1012.34C1026.04 1023.07 1028.57 1030.89 1033.63 1035.8 1038.72 1040.72 1046.76 1043.18 1057.76 1043.18H1072.98"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1171.77 961.336V954.004H1096.68C1097.39 942.539 1100.73 933.801 1106.72 927.785 1112.71 921.797 1121.05 918.805 1131.73 918.805 1137.94 918.805 1143.95 919.586 1149.77 921.148 1155.59 922.715 1161.36 925.063 1167.07 928.191V914.109C1161.31 911.055 1155.39 908.719 1149.33 907.105 1143.29 905.52 1137.17 904.723 1130.96 904.723 1115.36 904.723 1103.01 909.281 1093.89 918.398 1084.8 927.543 1080.25 939.898 1080.25 955.469 1080.25 971.578 1084.61 984.352 1093.34 993.785 1102.07 1003.25 1113.84 1007.98 1128.65 1007.98 1141.95 1007.98 1152.46 1003.8 1160.18 995.438 1167.91 987.078 1171.77 975.711 1171.77 961.336M1155.34 965.738C1155.22 974.293 1152.71 981.125 1147.82 986.234 1142.96 991.34 1136.52 993.895 1128.5 993.895 1119.41 993.895 1112.13 991.426 1106.65 986.488 1101.2 981.551 1098.06 974.598 1097.23 965.625L1155.34 965.738"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1255.47 991.551C1253.71 992.355 1251.8 992.941 1249.75 993.309 1247.7 993.699 1245.42 993.895 1242.93 993.895 1234.13 993.895 1227.37 990.949 1222.65 985.059 1217.94 979.191 1215.58 970.758 1215.58 959.762V907.07H1199.15V1005.63H1215.58V991.551C1218.85 997.098 1223.12 1001.22 1228.38


1003.91 1233.63 1006.62 1240.01 1007.98 1247.51 1007.98 1248.59 1007.98 1249.77 1007.98 1251.07 1007.98 1252.37 1007.98 1253.81 1007.98 1255.4 1007.98L1255.47 991.551"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1370.73 1043.18C1362.78 1029.75 1356.88 1016.48 1353.02 1003.36 1349.18 990.254 1347.26 976.957 1347.26 963.465 1347.26 949.996 1349.2 936.648 1353.09 923.422 1356.98 910.223 1362.86 896.949 1370.73 883.605H1356.5C1347.92 897.316 1341.49 910.785 1337.21 924.012 1332.96 937.258 1330.84 950.41 1330.84 963.465 1330.84 976.469 1332.95 989.559 1337.18 1002.73 1341.41 1015.93 1347.85 1029.41 1356.5 1043.18H1370.73"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1450.91 1020.85 1408.3 951.656H1450.91V1020.85M1447.87 1036.14H1469.69V951.656H1488.46V937.578H1469.69V907.07H1450.91V937.578H1394.59V954.406L1447.87 1036.14"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1535.81 970.43V921.148H1562.79C1571.96 921.148 1578.75 923.18 1583.14 927.238 1587.57 931.293 1589.78 937.492 1589.78 945.828 1589.78 954.234 1587.57 960.434 1583.14 964.418 1578.75 968.426 1571.96 970.43 1562.79 970.43H1535.81M1535.81 1022.05V984.508H1560.7C1568.92 984.508 1575.03 986.051 1579.04 989.129 1583.07 992.234 1585.09 996.953 1585.09 1003.28 1585.09 1009.57 1583.07 1014.26 1579.04 1017.36 1575.03 1020.49 1568.92 1022.05 1560.7 1022.05H1535.81M1517.04 1036.14H1561.95C1575.37 1036.14 1585.71 1033.45 1592.97 1028.07 1600.23 1022.69 1603.86 1015.04 1603.86 1005.12 1603.86 997.441 1601.93 991.328 1598.07 986.781 1594.21 982.262 1588.52 979.438 1581.02 978.313 1589.72 976.43 1596.48 972.496 1601.29 966.508 1606.13 960.52 1608.55 953.039 1608.55 944.066 1608.55 932.262 1604.66 923.141 1596.86 916.715 1589.06 910.285 1577.98 907.07 1563.6 907.07H1517.04V1036.14"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1634.6 963.391H1681.54V949.309H1634.6V963.391"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1747.84 979.816C1740.27 979.816 1734.26 977.09 1729.84 971.641 1725.41 966.215 1723.2 958.77 1723.2 949.309 1723.2 939.898 1725.41 932.457 1729.84 926.98 1734.26 921.527 1740.27 918.805 1747.84 918.805 1755.42 918.805 1761.42 921.527 1765.84 926.98 1770.27 932.457 1772.48 939.898 1772.48 949.309 1772.48 958.77 1770.27 966.215 1765.84 971.641 1761.42 977.09 1755.42 979.816 1747.84 979.816M1781.87 1033.79V1017.36C1777.64 1019.64 1773.38 1021.37 1769.07 1022.57 1764.79 1023.79 1760.54 1024.4 1756.31 1024.4 1745.21 1024.4 1736.73 1020.55 1730.86 1012.85 1725.02 1005.18 1721.69 993.578 1720.86 978.055 1724.18 983.164 1728.36 987.078 1733.39 989.789 1738.43 992.527 1743.98 993.895 1750.04 993.895 1762.75 993.895 1772.8 989.898 1780.18 981.906 1787.56 973.938 1791.25 963.07 1791.25 949.309 1791.25 935.84 1787.33 925.035 1779.48 916.898 1771.64 908.781 1761.2 904.723 1748.17 904.723 1733.23 904.723 1721.82 910.43 1713.93 921.848 1706.03 933.285 1702.08 949.859 1702.08 971.566 1702.08 991.953 1706.86 1008.2 1716.42 1020.3 1726 1032.42 1738.85 1038.48 1754.95 1038.48 1759.28 1038.48 1763.64 1038.09 1768.04 1037.31 1772.47 1036.53 1777.08 1035.36 1781.87 1033.79"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1871.46 1020.85 1828.85 951.656H1871.46V1020.85M1868.41 1036.14H1890.23V951.656H1909V937.578H1890.23V907.07H1871.46V937.578H1815.14V954.406L1868.41 1036.14"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1956.35 970.43V921.148H1983.34C1992.5 921.148 1999.29 923.18 2003.69 927.238 2008.11 931.293 2010.32 937.492 2010.32 945.828 2010.32 954.234 2008.11 960.434 2003.69 964.418 1999.29 968.426 1992.5 970.43 1983.34 970.43H1956.35M1956.35 1022.05V984.508H1981.25C1989.46 984.508 1995.57 986.051 1999.58 989.129 2003.62 992.234 2005.63 996.953 2005.63 1003.28 2005.63 1009.57 2003.62 1014.26 1999.58 1017.36 1995.57 1020.49 1989.46 1022.05 1981.25 1022.05H1956.35M1937.58 1036.14H1982.5C1995.91 1036.14 2006.25 1033.45 2013.52 1028.07 2020.77 1022.69 2024.41 1015.04 2024.41 1005.12 2024.41 997.441 2022.47 991.328 2018.61 986.781 2014.75 982.262 2009.07 979.438 2001.56 978.313 2010.27 976.43 2017.02 972.496 2021.84 966.508 2026.68 960.52 2029.1 953.039 2029.1 944.066 2029.1 932.262 2025.2 923.141 2017.4 916.715 2009.61 910.285 1998.52 907.07 1984.14 907.07H1937.58V1036.14"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M2059.84 1043.18H2074.07C2082.64 1029.41 2089.06 1015.93 2093.32 1002.73 2097.59 989.559 2099.73 976.469 2099.73 963.465 2099.73 950.41 2097.59 937.258 2093.32 924.012 2089.06 910.785 2082.64 897.316 2074.07 883.605H2059.84C2067.71 896.949 2073.59 910.223 2077.48 923.422 2081.36 936.648 2083.3

949.996 2083.3 963.465 2083.3 976.957 2081.36 990.254 2077.48 1003.36 2073.59 1016.48 2067.71 1029.75 2059.84 1043.18"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M622.203 4294.7H2443.283V4901.727H622.203Z" fill="#ffffff" fill-rule="evenodd"/> <path transform="matrix(.1,0,0,-.1,0,796)" stroke-width="2.8346" stroke-linecap="butt" stroke-miterlimit="10" stroke-linejoin="miter" fill="none" stroke="#ffffff" d="M622.203 4294.7H2443.283V4901.727H622.203Z"/> <path transform="matrix(.1,0,0,-.1,0,796)" stroke-width="30.3515" stroke-linecap="butt" stroke-miterlimit="10" stroke-linejoin="miter" fill="none" stroke="#000000" d="M622.203 4294.7H2443.283V4901.727H622.203Z"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1171.79 4727.1C1170.04 4727.91 1168.13 4728.49 1166.07 4728.86 1164.02 4729.25 1161.75 4729.45 1159.25 4729.45 1150.45 4729.45 1143.7 4726.5 1138.98 4720.61 1134.26 4714.74 1131.9 4706.31 1131.9 4695.31V4642.62H1115.47V4741.18H1131.9V4727.1C1135.18 4732.65 1139.44 4736.77 1144.7 4739.46 1149.95 4742.17 1156.33 4743.52 1163.84 4743.52 1164.91 4743.52 1166.1 4743.52 1167.39 4743.52 1168.69 4743.52 1170.13 4743.52 1171.72 4743.52L1171.79 4727.1"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1270.04 4696.89V4689.55H1194.95C1195.66 4678.09 1199.01 4669.35 1205 4663.34 1210.98 4657.35 1219.32 4654.35 1230 4654.35 1236.21 4654.35 1242.22 4655.14 1248.04 4656.7 1253.86 4658.27 1259.63 4660.61 1265.35 4663.74V4649.66C1259.58 4646.61 1253.66 4644.27 1247.6 4642.66 1241.56 4641.07 1235.44 4640.27 1229.23 4640.27 1213.64 4640.27 1201.28 4644.83 1192.16 4653.95 1183.07 4663.09 1178.52 4675.45 1178.52 4691.02 1178.52 4707.13 1182.89 4719.9 1191.61 4729.34 1200.34 4738.8 1212.11 4743.52 1226.92 4743.52 1240.22 4743.52 1250.73 4739.34 1258.45 4730.98 1266.18 4722.63 1270.04 4711.26 1270.04 4696.89M1253.61 4701.29C1253.49 4709.84 1250.99 4716.67 1246.1 4721.78 1241.23 4726.89 1234.79 4729.45 1226.77 4729.45 1217.68 4729.45 1210.4 4726.98 1204.92 4722.04 1199.47 4717.1 1196.33 4710.15 1195.5 4701.18L1253.61 4701.29"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1358.44 4736.48V4722.41C1354.04 4724.75 1349.46 4726.51 1344.72 4727.68 1339.98 4728.86 1335.07 4729.45 1329.98 4729.45 1322.26 4729.45 1316.46 4728.23 1312.6 4725.82 1308.74 4723.39 1306.81 4719.77 1306.81 4714.93 1306.81 4711.23 1308.15 4708.34 1310.84 4706.23 1313.55 4704.13 1319.01 4702.13 1327.2 4700.22L1332.44 4699.01C1343.54 4696.57 1351.42 4693.11 1356.09 4688.64 1360.78 4684.16 1363.13 4677.92 1363.13 4669.9 1363.13 4660.78 1359.63 4653.56 1352.64 4648.23 1345.65 4642.93 1336.03 4640.27 1323.79 4640.27 1318.68 4640.27 1313.36 4640.86 1307.84 4642.03 1302.31 4643.21 1296.5 4644.96 1290.38 4647.31V4663.74C1296.13 4660.61 1301.79 4658.27 1307.36 4656.7 1312.93 4655.14 1318.45 4654.35 1323.89 4654.35 1331.23 4654.35 1336.86 4655.64 1340.8 4658.2 1344.73 4660.79 1346.7 4664.42 1346.7 4669.09 1346.7 4673.45 1345.31 4676.77 1342.52 4679.07 1339.73 4681.39 1333.61 4683.63 1324.15 4685.78L1318.84 4687.06C1308.79 4689.16 1301.53 4692.39 1297.05 4696.74 1292.61 4701.09 1290.38 4707.05 1290.38 4714.63 1290.38 4723.85 1293.61 4730.96 1300.06 4735.97 1306.52 4741.01 1315.67 4743.52 1327.53 4743.52 1333.39 4743.52 1338.92 4742.94 1344.1 4741.77 1349.28 4740.59 1354.06 4738.83 1358.44 4736.48"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1476.06 4696.89V4689.55H1400.97C1401.68 4678.09 1405.03 4669.35 1411.02 4663.34 1417.01 4657.35 1425.34 4654.35 1436.02 4654.35 1442.23 4654.35 1448.25 4655.14 1454.06 4656.7 1459.88 4658.27 1465.65 4660.61 1471.37 4663.74V4649.66C1465.6 4646.61 1459.69 4644.27 1453.63 4642.66 1447.59 4641.07 1441.46 4640.27 1435.25 4640.27 1419.66 4640.27 1407.3 4644.83 1398.18 4653.95 1389.09 4663.09 1384.54 4675.45 1384.54 4691.02 1384.54 4707.13 1388.91 4719.9 1397.63 4729.34 1406.36 4738.8 1418.13 4743.52 1432.95 4743.52 1446.24 4743.52 1456.75 4739.34 1464.48 4730.98 1472.2 4722.63 1476.06 4711.26 1476.06 4696.89M1459.64 4701.29C1459.52 4709.84 1457.01 4716.67 1452.12 4721.78 1447.26 4726.89 1440.82 4729.45 1432.8 4729.45 1423.7 4729.45 1416.42 4726.98 1410.95 4722.04 1405.49 4717.1 1402.35 4710.15 1401.52 4701.18L1459.64 4701.29"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1519.87 4769.34V4741.18H1552.72V4729.45H1519.87V4675.69C1519.87 4667.63 1520.95 4662.45 1523.1 4660.14 1525.27 4657.85 1529.69 4656.7 1536.34 4656.7H1552.72V4642.62H1536.34C1523.89 4642.62 1515.3 4644.99 1510.56 4649.73 1505.82 4654.5 1503.45 4663.15 1503.45 4675.69V4729.45H1491.71V4741.18H1503.45V4769.34H1519.87"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1603.8 4771.68H1618.83L1572.85 4626.19H1557.82L1603.8 4771.68"/> <path

transform="matrix(.1,0,0,-.1,0,796)" d="M1717.26 4702.97V4642.62H1700.84V4702.46C1700.84 4711.48 1699.04 4718.23 1695.45 4722.7 1691.88 4727.2 1686.51 4729.45 1679.35 4729.45 1670.75 4729.45 1663.96 4726.73 1659 4721.3 1654.04 4715.9 1651.55 4708.52 1651.55 4699.16V4642.62H1635.13V4778.73H1651.55V4727.1C1655.39 4732.6 1659.91 4736.71 1665.12 4739.42 1670.33 4742.16 1676.33 4743.52 1683.13 4743.52 1694.37 4743.52 1702.86 4740.09 1708.61 4733.22 1714.38 4726.38 1717.26 4716.29 1717.26 4702.97"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1792.87 4691.9C1780.2 4691.9 1771.43 4690.45 1766.54 4687.54 1761.65 4684.63 1759.21 4679.65 1759.21 4672.61 1759.21 4667.02 1761.04 4662.57 1764.71 4659.27 1768.38 4655.99 1773.36 4654.35 1779.67 4654.35 1788.37 4654.35 1795.35 4657.43 1800.61 4663.59 1805.86 4669.78 1808.49 4677.99 1808.49 4688.23V4691.9H1792.87M1824.91 4698.09V4642.62H1808.49V4656.7C1804.87 4651.08 1800.36 4646.93 1794.96 4644.27 1789.55 4641.61 1782.94 4640.27 1775.12 4640.27 1765.22 4640.27 1757.35 4643.04 1751.51 4648.56 1745.69 4654.11 1742.78 4661.53 1742.78 4670.82 1742.78 4681.67 1746.43 4689.84 1753.75 4695.34 1761.05 4700.87 1771.97 4703.63 1786.49 4703.63H1808.49V4705.39C1808.49 4713.04 1806.08 4718.96 1801.27 4723.14 1796.47 4727.34 1789.74 4729.45 1781.06 4729.45 1775.54 4729.45 1770.16 4728.86 1764.93 4727.68 1759.7 4726.51 1754.66 4724.75 1749.82 4722.41V4736.48C1755.66 4738.83 1761.34 4740.59 1766.84 4741.77 1772.36 4742.94 1777.72 4743.52 1782.93 4743.52 1797.01 4743.52 1807.52 4739.76 1814.46 4732.23 1821.43 4724.73 1824.91 4713.35 1824.91 4698.09"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1860.97 4778.73H1877.4V4642.62H1860.97V4778.73"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1927.66 4769.34V4741.18H1960.52V4729.45H1927.66V4675.69C1927.66 4667.63 1928.74 4662.45 1930.89 4660.14 1933.07 4657.85 1937.48 4656.7 1944.13 4656.7H1960.52V4642.62H1944.13C1931.69 4642.62 1923.09 4644.99 1918.35 4649.73 1913.61 4654.5 1911.24 4663.15 1911.24 4675.69V4729.45H1899.51V4741.18H1911.24V4769.34H1927.66"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1305.28 4509.72V4495.64C1300.68 4497.98 1296.08 4499.75 1291.46 4500.92 1286.86 4502.09 1282.22 4502.68 1277.52 4502.68 1266.99 4502.68 1258.81 4499.39 1253 4492.81 1247.18 4486.24 1244.27 4477.01 1244.27 4465.13 1244.27 4453.25 1247.18 4444.02 1253 4437.45 1258.81 4430.87 1266.99 4427.59 1277.52 4427.59 1282.22 4427.59 1286.86 4428.17 1291.46 4429.34 1296.08 4430.52 1300.68 4432.28 1305.28 4434.63V4420.55C1300.78 4418.2 1296.13 4416.44 1291.31 4415.27 1286.52 4414.09 1281.41 4413.5 1275.98 4413.5 1261.25 4413.5 1249.54 4418.15 1240.86 4427.44 1232.18 4436.75 1227.84 4449.32 1227.84 4465.13 1227.84 4481.17 1232.23 4493.78 1241 4502.97 1249.78 4512.16 1261.79 4516.76 1277.05 4516.76 1282.01 4516.76 1286.85 4516.17 1291.57 4515 1296.29 4513.82 1300.86 4512.06 1305.28 4509.72"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1372.29 4502.68C1363.59 4502.68 1356.71 4499.33 1351.65 4492.63 1346.61 4485.93 1344.09 4476.77 1344.09 4465.13 1344.09 4453.5 1346.6 4444.33 1351.61 4437.63 1356.64 4430.93 1363.54 4427.59 1372.29 4427.59 1380.95 4427.59 1387.79 4430.95 1392.82 4437.67 1397.88 4444.39 1400.41 4453.54 1400.41 4465.13 1400.41 4476.67 1397.88 4485.81 1392.82 4492.56 1387.79 4499.3 1380.95 4502.68 1372.29 4502.68M1372.25 4516.76C1386.14 4516.76 1397.04 4512.19 1404.96 4503.04 1412.88 4493.93 1416.84 4481.29 1416.84 4465.13 1416.84 4449.02 1412.88 4436.39 1404.96 4427.22 1397.04 4418.08 1386.14 4413.5 1372.25 4413.5 1358.32 4413.5 1347.41 4418.08 1339.51 4427.22 1331.61 4436.39 1327.67 4449.02 1327.67 4465.13 1327.67 4481.29 1331.61 4493.93 1339.51 4503.04 1347.41 4512.19 1358.32 4516.76 1372.25 4516.76"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1527.99 4476.2V4415.85H1511.57V4475.69C1511.57 4484.71 1509.77 4491.46 1506.18 4495.93 1502.61 4500.43 1497.24 4502.68 1490.08 4502.68 1481.48 4502.68 1474.69 4499.96 1469.73 4494.54 1464.77 4489.14 1462.29 4481.75 1462.29 4472.39V4415.85H1445.86V4514.41H1462.29V4500.33C1466.13 4505.83 1470.65 4509.94 1475.85 4512.65 1481.06 4515.39 1487.06 4516.76 1493.86 4516.76 1505.1 4516.76 1513.59 4513.32 1519.34 4506.45 1525.11 4499.61 1527.99 4489.53 1527.99 4476.2"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1576.98 4542.57V4514.41H1609.83V4502.68H1576.98V4448.93C1576.98 4440.86 1578.05 4435.68 1580.21 4433.38 1582.38 4431.08 1586.79 4429.93 1593.44 4429.93H1609.83V4415.85H1593.44C1581 4415.85 1572.41 4418.22 1567.66 4422.96 1562.92 4427.73 1560.55 4436.39 1560.55 4448.93V4502.68H1548.82V4514.41H1560.55V4542.57H1576.98"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1687.67 4500.33C1685.91 4501.14 1684 4501.72 1681.95

4502.09 1679.9 4502.48 1677.63 4502.68 1675.13 4502.68 1666.33 4502.68 1659.57 4499.73 1654.86 4493.84 1650.14 4487.97 1647.78 4479.54 1647.78 4468.54V4415.85H1631.35V4514.41H1647.78V4500.33C1651.05 4505.88 1655.32 4510 1660.57 4512.69 1665.83 4515.4 1672.21 4516.76 1679.71 4516.76 1680.79 4516.76 1681.98 4516.76 1683.27 4516.76 1684.57 4516.76 1686.01 4516.76 1687.6 4516.76L1687.67 4500.33"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1739.02 4502.68C1730.32 4502.68 1723.44 4499.33 1718.38 4492.63 1713.34 4485.93 1710.83 4476.77 1710.83 4465.13 1710.83 4453.5 1713.33 4444.33 1718.34 4437.63 1723.38 4430.93 1730.27 4427.59 1739.02 4427.59 1747.68 4427.59 1754.52 4430.95 1759.55 4437.67 1764.62 4444.39 1767.14 4453.54 1767.14 4465.13 1767.14 4476.67 1764.62 4485.81 1759.55 4492.56 1754.52 4499.3 1747.68 4502.68 1739.02 4502.68M1738.98 4516.76C1752.87 4516.76 1763.77 4512.19 1771.69 4503.04 1779.61 4493.93 1783.57 4481.29 1783.57 4465.13 1783.57 4449.02 1779.61 4436.39 1771.69 4427.22 1763.77 4418.08 1752.87 4413.5 1738.98 4413.5 1725.05 4413.5 1714.14 4418.08 1706.24 4427.22 1698.35 4436.39 1694.4 4449.02 1694.4 4465.13 1694.4 4481.29 1698.35 4493.93 1706.24 4503.04 1714.14 4512.19 1725.05 4516.76 1738.98 4516.76"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1812.59 4551.96H1829.02V4415.85H1812.59V4551.96"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M622.203 3384.17H2443.283V3991.1968H622.203Z" fill="#ffffff" fill-rule="evenodd"/> <path transform="matrix(.1,0,0,-.1,0,796)" stroke-width="2.8346" stroke-linecap="butt" stroke-miterlimit="10" stroke-linejoin="miter" fill="none" stroke="#ffffff" d="M622.203 3384.17H2443.283V3991.1968H622.203Z"/> <path transform="matrix(.1,0,0,-.1,0,796)" stroke-width="30.3515" stroke-linecap="butt" stroke-miterlimit="10" stroke-linejoin="miter" fill="none" stroke="#000000" d="M622.203 3384.17H2443.283V3991.1968H622.203Z"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1218.05 3781.37C1205.38 3781.37 1196.61 3779.91 1191.72 3777 1186.83 3774.1 1184.39 3769.12 1184.39 3762.08 1184.39 3756.48 1186.22 3752.04 1189.89 3748.73 1193.55 3745.46 1198.54 3743.82 1204.85 3743.82 1213.55 3743.82 1220.53 3746.9 1225.79 3753.06 1231.04 3759.25 1233.67 3767.46 1233.67 3777.7V3781.37H1218.05M1250.09 3787.57V3732.09H1233.67V3746.17C1230.05 3740.55 1225.54 3736.4 1220.14 3733.74 1214.73 3731.07 1208.12 3729.74 1200.3 3729.74 1190.4 3729.74 1182.53 3732.5 1176.69 3738.03 1170.87 3743.58 1167.96 3751 1167.96 3760.29 1167.96 3771.14 1171.61 3779.32 1178.92 3784.82 1186.23 3790.34 1197.15 3793.1 1211.67 3793.1H1233.67V3794.86C1233.67 3802.51 1231.26 3808.43 1226.44 3812.61 1221.65 3816.81 1214.92 3818.91 1206.24 3818.91 1200.71 3818.91 1195.34 3818.33 1190.11 3817.16 1184.88 3815.98 1179.84 3814.22 1175 3811.88V3825.95C1180.84 3828.3 1186.51 3830.06 1192.01 3831.23 1197.54 3832.41 1202.9 3833 1208.11 3833 1222.19 3833 1232.7 3829.23 1239.64 3821.7 1246.61 3814.2 1250.09 3802.82 1250.09 3787.57"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1356.55 3781.37C1356.55 3793.08 1354.14 3802.26 1349.33 3808.91 1344.54 3815.58 1337.95 3818.91 1329.57 3818.91 1321.18 3818.91 1314.58 3815.58 1309.77 3808.91 1304.98 3802.26 1302.58 3793.08 1302.58 3781.37 1302.58 3769.66 1304.98 3760.47 1309.77 3753.8 1314.58 3747.15 1321.18 3743.82 1329.57 3743.82 1337.95 3743.82 1344.54 3747.15 1349.33 3753.8 1354.14 3760.47 1356.55 3769.66 1356.55 3781.37M1302.58 3816.57C1305.91 3822.12 1310.11 3826.23 1315.19 3828.93 1320.28 3831.64 1326.35 3833 1333.42 3833 1345.15 3833 1354.67 3828.25 1361.98 3818.77 1369.31 3809.29 1372.98 3796.82 1372.98 3781.37 1372.98 3765.92 1369.31 3753.45 1361.98 3743.97 1354.67 3734.48 1345.15 3729.74 1333.42 3729.74 1326.35 3729.74 1320.28 3731.09 1315.19 3733.78 1310.11 3736.49 1305.91 3740.62 1302.58 3746.17V3732.09H1286.15V3868.2H1302.58V3816.57"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1462.57 3825.95V3811.88C1458.17 3814.22 1453.6 3815.98 1448.86 3817.16 1444.11 3818.33 1439.2 3818.91 1434.11 3818.91 1426.39 3818.91 1420.6 3817.7 1416.73 3815.29 1412.87 3812.86 1410.94 3809.23 1410.94 3804.39 1410.94 3800.7 1412.29 3797.81 1414.98 3795.7 1417.69 3793.6 1423.14 3791.6 1431.33 3789.69L1436.57 3788.48C1447.67 3786.04 1455.55 3782.58 1460.22 3778.11 1464.91 3773.63 1467.26 3767.39 1467.26 3759.37 1467.26 3750.25 1463.77 3743.03 1456.77 3737.7 1449.79 3732.39 1440.16 3729.74 1427.92 3729.74 1422.81 3729.74 1417.49 3730.33 1411.97 3731.5 1406.45 3732.68 1400.63 3734.44 1394.52 3736.78V3753.21C1400.26 3750.08 1405.92 3747.73 1411.49 3746.17 1417.07 3744.61 1422.58 3743.82 1428.03 3743.82 1435.36 3743.82 1441 3745.11 1444.93 3747.67 1448.87 3750.26 1450.84 3753.89 1450.84 3758.56 1450.84 3762.91 1449.44 3766.24 1446.66 3768.54

1443.87 3770.86 1437.75 3773.09 1428.29 3775.25L1422.97 3776.53C1412.92 3778.63 1405.66 3781.86 1401.19 3786.21 1396.74 3790.56 1394.52 3796.52 1394.52 3804.1 1394.52 3813.32 1397.74 3820.43 1404.2 3825.44 1410.65 3830.48 1419.8 3833 1431.66 3833 1437.52 3833 1443.05 3832.41 1448.23 3831.23 1453.41 3830.06 1458.19 3828.3 1462.57 3825.95"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1512.14 3858.81V3830.65H1545V3818.91H1512.14V3765.16C1512.14 3757.09 1513.22 3751.91 1515.37 3749.62 1517.55 3747.32 1521.96 3746.17 1528.61 3746.17H1545V3732.09H1528.61C1516.16 3732.09 1507.57 3734.46 1502.83 3739.2 1498.09 3743.97 1495.71 3752.62 1495.71 3765.16V3818.91H1483.98V3830.65H1495.71V3858.81H1512.14"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1622.84 3816.57C1621.07 3817.38 1619.17 3817.96 1617.11 3818.33 1615.06 3818.72 1612.79 3818.91 1610.29 3818.91 1601.5 3818.91 1594.73 3815.97 1590.02 3810.08 1585.3 3804.21 1582.94 3795.78 1582.94 3784.78V3732.09H1566.52V3830.65H1582.94V3816.57C1586.22 3822.12 1590.48 3826.23 1595.74 3828.93 1600.99 3831.64 1607.38 3833 1614.88 3833 1615.95 3833 1617.14 3833 1618.43 3833 1619.73 3833 1621.17 3833 1622.76 3833L1622.84 3816.57"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1683.9 3781.37C1671.24 3781.37 1662.46 3779.91 1657.57 3777 1652.68 3774.1 1650.24 3769.12 1650.24 3762.08 1650.24 3756.48 1652.07 3752.04 1655.74 3748.73 1659.41 3745.46 1664.39 3743.82 1670.7 3743.82 1679.4 3743.82 1686.38 3746.9 1691.64 3753.06 1696.89 3759.25 1699.52 3767.46 1699.52 3777.7V3781.37H1683.9M1715.95 3787.57V3732.09H1699.52V3746.17C1695.9 3740.55 1691.39 3736.4 1685.99 3733.74 1680.59 3731.07 1673.98 3729.74 1666.15 3729.74 1656.25 3729.74 1648.38 3732.5 1642.54 3738.03 1636.72 3743.58 1633.81 3751 1633.81 3760.29 1633.81 3771.14 1637.47 3779.32 1644.78 3784.82 1652.09 3790.34 1663 3793.1 1677.52 3793.1H1699.52V3794.86C1699.52 3802.51 1697.11 3808.43 1692.3 3812.61 1687.5 3816.81 1680.77 3818.91 1672.09 3818.91 1666.57 3818.91 1661.19 3818.33 1655.96 3817.16 1650.73 3815.98 1645.69 3814.22 1640.85 3811.88V3825.95C1646.7 3828.3 1652.37 3830.06 1657.87 3831.23 1663.39 3832.41 1668.76 3833 1673.96 3833 1688.04 3833 1698.55 3829.23 1705.5 3821.7 1712.46 3814.2 1715.95 3802.82 1715.95 3787.57"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1822.41 3825.95V3811.88C1817.81 3814.22 1813.2 3815.98 1808.58 3817.16 1803.99 3818.33 1799.34 3818.91 1794.65 3818.91 1784.11 3818.91 1775.94 3815.63 1770.12 3809.05 1764.3 3802.48 1761.39 3793.25 1761.39 3781.37 1761.39 3769.49 1764.3 3760.26 1770.12 3753.68 1775.94 3747.11 1784.11 3743.82 1794.65 3743.82 1799.34 3743.82 1803.99 3744.41 1808.58 3745.58 1813.2 3746.75 1817.81 3748.52 1822.41 3750.86V3736.78C1817.91 3734.44 1813.25 3732.68 1808.43 3731.5 1803.64 3730.33 1798.54 3729.74 1793.11 3729.74 1778.37 3729.74 1766.66 3734.39 1757.98 3743.68 1749.3 3752.99 1744.96 3765.55 1744.96 3781.37 1744.96 3797.4 1749.36 3810.02 1758.13 3819.21 1766.91 3828.4 1778.92 3833 1794.17 3833 1799.13 3833 1803.97 3832.41 1808.69 3831.23 1813.41 3830.06 1817.98 3828.3 1822.41 3825.95"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1868.26 3858.81V3830.65H1901.11V3818.91H1868.26V3765.16C1868.26 3757.09 1869.34 3751.91 1871.48 3749.62 1873.66 3747.32 1878.07 3746.17 1884.72 3746.17H1901.11V3732.09H1884.72C1872.28 3732.09 1863.69 3734.46 1858.95 3739.2 1854.2 3743.97 1851.83 3752.62 1851.83 3765.16V3818.91H1840.1V3830.65H1851.83V3858.81H1868.26"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1119.61 3599.19V3585.11C1115.02 3587.45 1110.41 3589.21 1105.79 3590.39 1101.2 3591.56 1096.55 3592.15 1091.86 3592.15 1081.32 3592.15 1073.14 3588.86 1067.33 3582.29 1061.51 3575.71 1058.6 3566.48 1058.6 3554.6 1058.6 3542.72 1061.51 3533.49 1067.33 3526.92 1073.14 3520.34 1081.32 3517.05 1091.86 3517.05 1096.55 3517.05 1101.2 3517.64 1105.79 3518.82 1110.41 3519.99 1115.02 3521.75 1119.61 3524.09V3510.02C1115.12 3507.67 1110.46 3505.91 1105.64 3504.73 1100.86 3503.56 1095.75 3502.98 1090.32 3502.98 1075.58 3502.98 1063.87 3507.62 1055.19 3516.91 1046.52 3526.22 1042.18 3538.79 1042.18 3554.6 1042.18 3570.64 1046.56 3583.25 1055.34 3592.44 1064.11 3601.63 1076.13 3606.23 1091.38 3606.23 1096.34 3606.23 1101.18 3605.64 1105.9 3604.47 1110.62 3603.29 1115.19 3601.54 1119.61 3599.19"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1186.63 3592.15C1177.92 3592.15 1171.04 3588.8 1165.98 3582.1 1160.95 3575.4 1158.43 3566.24 1158.43 3554.6 1158.43 3542.96 1160.93 3533.8 1165.95 3527.1 1170.98 3520.4 1177.87 3517.05 1186.63 3517.05 1195.28 3517.05 1202.12 3520.41 1207.16 3527.14 1212.22 3533.86 1214.75 3543.02 1214.75 3554.6 1214.75 3566.14 1212.22 3575.28 1207.16 3582.03 1202.12 3588.77 1195.28 3592.15 1186.63

3592.15M1186.59 3606.23C1200.47 3606.23 1211.38 3601.66 1219.29 3592.52 1227.21 3583.39 1231.17 3570.76 1231.17 3554.6 1231.17 3538.49 1227.21 3525.86 1219.29 3516.69 1211.38 3507.55 1200.47 3502.98 1186.59 3502.98 1172.65 3502.98 1161.74 3507.55 1153.84 3516.69 1145.95 3525.86 1142 3538.49 1142 3554.6 1142 3570.76 1145.95 3583.39 1153.84 3592.52 1161.74 3601.66 1172.65 3606.23 1186.59 3606.23"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1337.23 3586.13C1341.26 3593 1346.07 3598.06 1351.64 3601.31 1357.24 3604.59 1363.84 3606.23 1371.44 3606.23 1381.63 3606.23 1389.49 3602.71 1395.02 3595.67 1400.57 3588.65 1403.34 3578.65 1403.34 3565.68V3505.32H1386.91V3565.16C1386.91 3574.3 1385.26 3581.09 1381.96 3585.51 1378.66 3589.93 1373.61 3592.15 1366.82 3592.15 1358.53 3592.15 1351.98 3589.43 1347.17 3584.01 1342.38 3578.61 1339.98 3571.22 1339.98 3561.86V3505.32H1323.55V3565.16C1323.55 3574.35 1321.9 3581.15 1318.6 3585.55 1315.3 3589.95 1310.21 3592.15 1303.31 3592.15 1295.13 3592.15 1288.62 3589.42 1283.81 3583.97 1279.02 3578.52 1276.62 3571.15 1276.62 3561.86V3505.32H1260.2V3603.88H1276.62V3589.8C1280.29 3595.42 1284.68 3599.57 1289.79 3602.23 1294.92 3604.89 1301 3606.23 1308.04 3606.23 1315.16 3606.23 1321.2 3604.5 1326.16 3601.06 1331.14 3597.64 1334.84 3592.66 1337.23 3586.13"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1513.52 3586.13C1517.55 3593 1522.35 3598.06 1527.93 3601.31 1533.52 3604.59 1540.13 3606.23 1547.73 3606.23 1557.92 3606.23 1565.78 3602.71 1571.3 3595.67 1576.85 3588.65 1579.63 3578.65 1579.63 3565.68V3505.32H1563.2V3565.16C1563.2 3574.3 1561.55 3581.09 1558.25 3585.51 1554.95 3589.93 1549.9 3592.15 1543.11 3592.15 1534.82 3592.15 1528.27 3589.43 1523.45 3584.01 1518.66 3578.61 1516.27 3571.22 1516.27 3561.86V3505.32H1499.84V3565.16C1499.84 3574.35 1498.19 3581.15 1494.89 3585.55 1491.59 3589.95 1486.49 3592.15 1479.6 3592.15 1471.41 3592.15 1464.91 3589.42 1460.09 3583.97 1455.3 3578.52 1452.91 3571.15 1452.91 3561.86V3505.32H1436.48V3603.88H1452.91V3589.8C1456.57 3595.42 1460.96 3599.57 1466.07 3602.23 1471.2 3604.89 1477.29 3606.23 1484.33 3606.23 1491.45 3606.23 1497.48 3604.5 1502.45 3601.06 1507.43 3597.64 1511.12 3592.66 1513.52 3586.13"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1655.81 3554.6C1643.15 3554.6 1634.38 3553.15 1629.49 3550.24 1624.6 3547.33 1622.16 3542.36 1622.16 3535.32 1622.16 3529.71 1623.99 3525.27 1627.66 3521.97 1631.32 3518.69 1636.31 3517.05 1642.61 3517.05 1651.32 3517.05 1658.3 3520.13 1663.55 3526.29 1668.8 3532.48 1671.43 3540.69 1671.43 3550.93V3554.6H1655.81M1687.86 3560.8V3505.32H1671.43V3519.4C1667.82 3513.78 1663.3 3509.64 1657.9 3506.97 1652.5 3504.31 1645.89 3502.98 1638.07 3502.98 1628.17 3502.98 1620.3 3505.74 1614.45 3511.26 1608.64 3516.81 1605.73 3524.23 1605.73 3533.52 1605.73 3544.37 1609.38 3552.55 1616.69 3558.05 1624 3563.57 1634.91 3566.34 1649.43 3566.34H1671.43V3568.09C1671.43 3575.75 1669.03 3581.66 1664.21 3585.84 1659.42 3590.05 1652.68 3592.15 1644.01 3592.15 1638.48 3592.15 1633.11 3591.56 1627.88 3590.39 1622.64 3589.21 1617.61 3587.45 1612.77 3585.11V3599.19C1618.61 3601.54 1624.28 3603.29 1629.78 3604.47 1635.3 3605.64 1640.67 3606.23 1645.88 3606.23 1659.96 3606.23 1670.47 3602.46 1677.41 3594.93 1684.38 3587.43 1687.86 3576.05 1687.86 3560.8"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1806.05 3565.68V3505.32H1789.63V3565.16C1789.63 3574.18 1787.83 3580.93 1784.24 3585.4 1780.67 3589.9 1775.3 3592.15 1768.14 3592.15 1759.54 3592.15 1752.75 3589.43 1747.79 3584.01 1742.83 3578.61 1740.35 3571.22 1740.35 3561.86V3505.32H1723.92V3603.88H1740.35V3589.8C1744.18 3595.3 1748.71 3599.41 1753.91 3602.12 1759.12 3604.86 1765.12 3606.23 1771.92 3606.23 1783.16 3606.23 1791.66 3602.79 1797.4 3595.93 1803.17 3589.08 1806.05 3579 1806.05 3565.68"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1901.97 3589.8V3641.43H1918.4V3505.32H1901.97V3519.4C1898.65 3513.85 1894.44 3509.72 1889.36 3507.01 1884.27 3504.32 1878.16 3502.98 1871.03 3502.98 1859.37 3502.98 1849.87 3507.72 1842.54 3517.2 1835.23 3526.69 1831.57 3539.15 1831.57 3554.6 1831.57 3570.05 1835.23 3582.52 1842.54 3592 1849.87 3601.48 1859.37 3606.23 1871.03 3606.23 1878.16 3606.23 1884.27 3604.87 1889.36 3602.16 1894.44 3599.47 1898.65 3595.35 1901.97 3589.8M1848 3554.6C1848 3542.89 1850.39 3533.7 1855.19 3527.03 1859.98 3520.38 1866.57 3517.05 1874.95 3517.05 1883.33 3517.05 1889.93 3520.38 1894.75 3527.03 1899.56 3533.7 1901.97 3542.89 1901.97 3554.6 1901.97 3566.31 1899.56 3575.49 1894.75 3582.14 1889.93 3588.81 1883.33 3592.15 1874.95 3592.15 1866.57


3592.15 1859.98 3588.81 1855.19 3582.14 1850.39 3575.49 1848 3566.31 1848 3554.6"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M2015.03 3599.19V3585.11C2010.63 3587.45 2006.05 3589.21 2001.31 3590.39 1996.57 3591.56 1991.66 3592.15 1986.57 3592.15 1978.85 3592.15 1973.06 3590.94 1969.2 3588.52 1965.33 3586.1 1963.4 3582.47 1963.4 3577.63 1963.4 3573.94 1964.75 3571.04 1967.43 3568.94 1970.15 3566.84 1975.6 3564.83 1983.79 3562.93L1989.03 3561.71C2000.13 3559.27 2008.01 3555.81 2012.68 3551.34 2017.38 3546.86 2019.72 3540.62 2019.72 3532.6 2019.72 3523.48 2016.23 3516.26 2009.23 3510.93 2002.24 3505.63 1992.63 3502.98 1980.38 3502.98 1975.27 3502.98 1969.95 3503.56 1964.43 3504.73 1958.9 3505.91 1953.09 3507.67 1946.98 3510.02V3526.44C1952.72 3523.31 1958.38 3520.96 1963.95 3519.4 1969.52 3517.84 1975.04 3517.05 1980.49 3517.05 1987.82 3517.05 1993.46 3518.34 1997.39 3520.91 2001.33 3523.5 2003.29 3527.13 2003.29 3531.79 2003.29 3536.14 2001.9 3539.47 1999.11 3541.77 1996.33 3544.09 1990.2 3546.33 1980.75 3548.48L1975.43 3549.76C1965.38 3551.86 1958.12 3555.09 1953.65 3559.44 1949.2 3563.79 1946.98 3569.76 1946.98 3577.34 1946.98 3586.55 1950.2 3593.66 1956.66 3598.68 1963.11 3603.71 1972.26 3606.23 1984.12 3606.23 1989.98 3606.23 1995.51 3605.64 2000.69 3604.47 2005.88 3603.29 2010.65 3601.54 2015.03 3599.19"/> <path transform="matrix(.1,0,0,-.1,0,796)" stroke-width="30.3515" stroke-linecap="butt" stroke-miterlimit="10" stroke-linejoin="miter" fill="none" stroke="#000000" d="M2443.28 3687.7H4723.63"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M4837.47 3687.7 4685.7 3611.82 4723.63 3687.7 4685.7 3763.55" fill-rule="evenodd"/> <path transform="matrix(.1,0,0,-.1,0,796)" stroke-width="2.8346" stroke-linecap="butt" stroke-miterlimit="10" stroke-linejoin="miter" fill="none" stroke="#000000" d="M4837.47 3687.7 4685.7 3611.82 4723.63 3687.7 4685.7 3763.55Z"/> <path transform="matrix(.1,0,0,-.1,0,796)" stroke-width="30.3515" stroke-linecap="butt" stroke-miterlimit="10" stroke-linejoin="miter" fill="none" stroke="#000000" d="M4837.47 3687.7 4685.7 3611.82 4723.63 3687.7 4685.7 3763.55Z"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M3353.81 5053.5H5174.89V5812.285H3353.81Z" fill="#ffffff" fill-rule="evenodd"/> <path transform="matrix(.1,0,0,-.1,0,796)" stroke-width="2.8346" stroke-linecap="butt" stroke-miterlimit="10" stroke-linejoin="miter" fill="none" stroke="#ffffff" d="M3353.81 5053.5H5174.89V5812.285H3353.81Z"/> <path transform="matrix(.1,0,0,-.1,0,796)" stroke-width="30.3515" stroke-linecap="butt" stroke-miterlimit="10" stroke-linejoin="miter" fill="none" stroke="#000000" d="M3353.81 5053.5H5174.89V5812.285H3353.81Z"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M3641.94 5592.27V5491.36H3663.06C3680.91 5491.36 3693.97 5495.42 3702.26 5503.53 3710.54 5511.65 3714.69 5524.46 3714.69 5541.96 3714.69 5559.34 3710.54 5572.06 3702.26 5580.13 3693.97 5588.22 3680.91 5592.27 3663.06 5592.27H3641.94M3623.17 5606.34H3660.39C3685.56 5606.34 3704.03 5601.14 3715.79 5590.73 3727.57 5580.34 3733.46 5564.08 3733.46 5541.96 3733.46 5519.69 3727.55 5503.34 3715.71 5492.9 3703.88 5482.48 3685.44 5477.28 3660.39 5477.28H3623.17V5606.34"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M3847.12 5531.55V5524.21H3772.03C3772.73 5512.75 3776.09 5504.01 3782.07 5498 3788.06 5492.01 3796.4 5489.01 3807.08 5489.01 3813.29 5489.01 3819.3 5489.79 3825.12 5491.36 3830.94 5492.92 3836.71 5495.27 3842.43 5498.4V5484.32C3836.66 5481.26 3830.74 5478.93 3824.68 5477.32 3818.64 5475.73 3812.52 5474.93 3806.31 5474.93 3790.71 5474.93 3778.36 5479.49 3769.24 5488.61 3760.15 5497.75 3755.6 5510.11 3755.6 5525.68 3755.6 5541.79 3759.96 5554.56 3768.69 5564 3777.42 5573.45 3789.19 5578.18 3804 5578.18 3817.3 5578.18 3827.81 5574 3835.53 5565.64 3843.26 5557.29 3847.12 5545.92 3847.12 5531.55M3830.69 5535.95C3830.57 5544.5 3828.07 5551.33 3823.18 5556.44 3818.31 5561.55 3811.87 5564.11 3803.85 5564.11 3794.76 5564.11 3787.48 5561.64 3782 5556.7 3776.55 5551.76 3773.41 5544.8 3772.58 5535.84L3830.69 5535.95"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M3944.9 5526.56C3944.9 5538.27 3942.49 5547.45 3937.68 5554.09 3932.89 5560.77 3926.3 5564.11 3917.91 5564.11 3909.53 5564.11 3902.93 5560.77 3898.11 5554.09 3893.32 5547.45 3890.93 5538.27 3890.93 5526.56 3890.93 5514.85 3893.32 5505.66 3898.11 5498.98 3902.93 5492.34 3909.53 5489.01 3917.91 5489.01 3926.3 5489.01 3932.89 5492.34 3937.68 5498.98 3942.49 5505.66 3944.9 5514.85 3944.9 5526.56M3890.93 5561.76C3894.25 5567.31 3898.46 5571.43 3903.54 5574.11 3908.63 5576.83 3914.7 5578.18 3921.77 5578.18 3933.5 5578.18 3943.02 5573.44 3950.33 5563.96 3957.66 5554.47 3961.33 5542.01 3961.33


5526.56 3961.33 5511.11 3957.66 5498.64 3950.33 5489.16 3943.02 5479.68 3933.5 5474.93 3921.77 5474.93 3914.7 5474.93 3908.63 5476.28 3903.54 5478.96 3898.46 5481.68 3894.25 5485.81 3890.93 5491.36V5477.28H3874.5V5613.38H3890.93V5561.76"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M3989.9 5515.41V5575.84H4006.33V5516.04C4006.33 5507.04 4008.11 5500.28 4011.68 5495.76 4015.25 5491.26 4020.62 5489.01 4027.78 5489.01 4036.36 5489.01 4043.14 5491.71 4048.13 5497.12 4053.12 5502.54 4055.61 5509.93 4055.61 5519.26V5575.84H4072.04V5477.28H4055.61V5491.36C4051.72 5485.81 4047.2 5481.68 4042.04 5478.96 4036.91 5476.28 4030.95 5474.93 4024.15 5474.93 4012.93 5474.93 4004.41 5478.37 3998.59 5485.23 3992.8 5492.13 3989.9 5502.19 3989.9 5515.41M4030.46 5578.18V5578.18"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M4167.95 5527.7C4167.95 5539.26 4165.58 5548.21 4160.84 5554.57 4156.13 5560.93 4149.49 5564.11 4140.93 5564.11 4132.43 5564.11 4125.8 5560.93 4121.06 5554.57 4116.34 5548.21 4113.98 5539.26 4113.98 5527.7 4113.98 5516.18 4116.34 5507.25 4121.06 5500.89 4125.8 5494.54 4132.43 5491.36 4140.93 5491.36 4149.49 5491.36 4156.13 5494.54 4160.84 5500.89 4165.58 5507.25 4167.95 5516.18 4167.95 5527.7M4184.38 5490C4184.38 5473.04 4180.7 5460.43 4173.34 5452.16 4166.01 5443.88 4154.78 5439.73 4139.65 5439.73 4134.03 5439.73 4128.73 5440.13 4123.77 5440.91 4118.81 5441.66 4113.98 5442.84 4109.29 5444.43V5460.85C4113.93 5458.46 4118.52 5456.68 4123.04 5455.54 4127.59 5454.39 4132.2 5453.81 4136.9 5453.81 4147.29 5453.81 4155.06 5456.56 4160.22 5462.06 4165.38 5467.56 4167.95 5475.86 4167.95 5486.96V5493.71C4164.68 5488.21 4160.49 5484.09 4155.38 5481.35 4150.27 5478.64 4144.14 5477.28 4137.01 5477.28 4125.18 5477.28 4115.64 5481.88 4108.41 5491.07 4101.17 5500.28 4097.55 5512.49 4097.55 5527.7 4097.55 5542.95 4101.17 5555.17 4108.41 5564.36 4115.64 5573.58 4125.18 5578.18 4137.01 5578.18 4144.14 5578.18 4150.27 5576.82 4155.38 5574.08 4160.49 5571.36 4164.68 5567.26 4167.95 5561.76V5575.84H4184.38V5490"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M4277.34 5606.34H4304.99L4337.14 5519.37 4369.48 5606.34H4397.02V5477.28H4378.25V5590.76L4345.76 5503.09H4328.6L4296.12 5590.76V5477.28H4277.34V5606.34"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M4471.39 5564.11C4462.69 5564.11 4455.81 5560.76 4450.75 5554.06 4445.71 5547.36 4443.2 5538.2 4443.2 5526.56 4443.2 5514.92 4445.7 5505.76 4450.71 5499.06 4455.75 5492.36 4462.64 5489.01 4471.39 5489.01 4480.04 5489.01 4486.89 5492.38 4491.92 5499.09 4496.98 5505.82 4499.51 5514.97 4499.51 5526.56 4499.51 5538.1 4496.98 5547.24 4491.92 5553.98 4486.89 5560.73 4480.04 5564.11 4471.39 5564.11M4471.35 5578.18C4485.24 5578.18 4496.14 5573.61 4504.06 5564.47 4511.98 5555.36 4515.94 5542.71 4515.94 5526.56 4515.94 5510.45 4511.98 5497.81 4504.06 5488.64 4496.14 5479.5 4485.24 5474.93 4471.35 5474.93 4457.42 5474.93 4446.51 5479.5 4438.61 5488.64 4430.71 5497.81 4426.77 5510.45 4426.77 5526.56 4426.77 5542.71 4430.71 5555.36 4438.61 5564.47 4446.51 5573.61 4457.42 5578.18 4471.35 5578.18"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M4608.32 5561.76V5613.38H4624.75V5477.28H4608.32V5491.36C4605 5485.81 4600.79 5481.68 4595.71 5478.96 4590.62 5476.28 4584.51 5474.93 4577.37 5474.93 4565.71 5474.93 4556.21 5479.68 4548.88 5489.16 4541.57 5498.64 4537.92 5511.11 4537.92 5526.56 4537.92 5542.01 4541.57 5554.47 4548.88 5563.96 4556.21 5573.44 4565.71 5578.18 4577.37 5578.18 4584.51 5578.18 4590.62 5576.83 4595.71 5574.11 4600.79 5571.43 4605 5567.31 4608.32 5561.76M4554.35 5526.56C4554.35 5514.85 4556.74 5505.66 4561.53 5498.98 4566.32 5492.34 4572.91 5489.01 4581.3 5489.01 4589.68 5489.01 4596.28 5492.34 4601.1 5498.98 4605.91 5505.66 4608.32 5514.85 4608.32 5526.56 4608.32 5538.27 4605.91 5547.45 4601.1 5554.09 4596.28 5560.77 4589.68 5564.11 4581.3 5564.11 4572.91 5564.11 4566.32 5560.77 4561.53 5554.09 4556.74 5547.45 4554.35 5538.27 4554.35 5526.56"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M4660.36 5515.41V5575.84H4676.79V5516.04C4676.79 5507.04 4678.57 5500.28 4682.14 5495.76 4685.71 5491.26 4691.07 5489.01 4698.24 5489.01 4706.82 5489.01 4713.6 5491.71 4718.59 5497.12 4723.57 5502.54 4726.07 5509.93 4726.07 5519.26V5575.84H4742.5V5477.28H4726.07V5491.36C4722.18 5485.81 4717.66 5481.68 4712.5 5478.96 4707.37 5476.28 4701.4 5474.93 4694.61 5474.93 4683.39 5474.93 4674.87 5478.37 4669.05 5485.23 4663.26 5492.13 4660.36 5502.19 4660.36 5515.41M4700.91 5578.18V5578.18"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M4775.05 5613.38H4791.48V5477.28H4775.05V5613.38"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M4909.8 5531.55V5524.21H4834.71C4835.41 5512.75


4838.76 5504.01 4844.75 5498 4850.74 5492.01 4859.08 5489.01 4869.76 5489.01 4875.97 5489.01 4881.98 5489.79 4887.8 5491.36 4893.62 5492.92 4899.39 5495.27 4905.11 5498.4V5484.32C4899.34 5481.26 4893.42 5478.93 4887.36 5477.32 4881.32 5475.73 4875.2 5474.93 4868.99 5474.93 4853.39 5474.93 4841.04 5479.49 4831.92 5488.61 4822.82 5497.75 4818.28 5510.11 4818.28 5525.68 4818.28 5541.79 4822.64 5554.56 4831.37 5564 4840.1 5573.45 4851.87 5578.18 4866.68 5578.18 4879.98 5578.18 4890.49 5574 4898.21 5565.64 4905.94 5557.29 4909.8 5545.92 4909.8 5531.55M4893.37 5535.95C4893.25 5544.5 4890.75 5551.33 4885.86 5556.44 4880.99 5561.55 4874.55 5564.11 4866.53 5564.11 4857.44 5564.11 4850.16 5561.64 4844.68 5556.7 4839.23 5551.76 4836.09 5544.8 4835.26 5535.84L4893.37 5535.95"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M3599.78 5379.58H3618.55V5250.51H3599.78V5379.58"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M3735.72 5310.86V5250.51H3719.3V5310.35C3719.3 5319.37 3717.5 5326.12 3713.91 5330.59 3710.34 5335.09 3704.97 5337.34 3697.81 5337.34 3689.21 5337.34 3682.42 5334.63 3677.46 5329.2 3672.5 5323.79 3670.02 5316.41 3670.02 5307.05V5250.51H3653.59V5349.07H3670.02V5334.99C3673.86 5340.49 3678.38 5344.6 3683.58 5347.31 3688.79 5350.05 3694.79 5351.42 3701.59 5351.42 3712.83 5351.42 3721.32 5347.98 3727.07 5341.11 3732.84 5334.27 3735.72 5324.19 3735.72 5310.86"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M3784.71 5377.23V5349.07H3817.56V5337.34H3784.71V5283.59C3784.71 5275.52 3785.79 5270.34 3787.93 5268.04 3790.11 5265.74 3794.52 5264.59 3801.17 5264.59H3817.56V5250.51H3801.17C3788.73 5250.51 3780.14 5252.88 3775.39 5257.63 3770.65 5262.39 3768.28 5271.04 3768.28 5283.59V5337.34H3756.55V5349.07H3768.28V5377.23H3784.71"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M3923.56 5304.78V5297.45H3848.47C3849.18 5285.98 3852.52 5277.24 3858.52 5271.23 3864.5 5265.24 3872.84 5262.25 3883.52 5262.25 3889.73 5262.25 3895.74 5263.03 3901.56 5264.59 3907.38 5266.16 3913.15 5268.5 3918.87 5271.63V5257.55C3913.1 5254.5 3907.18 5252.16 3901.12 5250.55 3895.08 5248.96 3888.96 5248.16 3882.75 5248.16 3867.16 5248.16 3854.8 5252.72 3845.68 5261.84 3836.59 5270.98 3832.04 5283.34 3832.04 5298.91 3832.04 5315.02 3836.4 5327.79 3845.13 5337.23 3853.86 5346.69 3865.63 5351.42 3880.44 5351.42 3893.74 5351.42 3904.25 5347.24 3911.97 5338.88 3919.7 5330.52 3923.56 5319.15 3923.56 5304.78M3907.13 5309.18C3907.01 5317.73 3904.5 5324.57 3899.62 5329.67 3894.75 5334.78 3888.31 5337.34 3880.29 5337.34 3871.2 5337.34 3863.91 5334.87 3858.44 5329.93 3852.99 5324.99 3849.85 5318.04 3849.02 5309.07L3907.13 5309.18"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M4007.26 5334.99C4005.5 5335.8 4003.59 5336.38 4001.54 5336.75 3999.49 5337.14 3997.21 5337.34 3994.72 5337.34 3985.92 5337.34 3979.16 5334.39 3974.45 5328.5 3969.73 5322.63 3967.37 5314.2 3967.37 5303.2V5250.51H3950.94V5349.07H3967.37V5334.99C3970.64 5340.54 3974.91 5344.66 3980.16 5347.35 3985.42 5350.06 3991.8 5351.42 3999.3 5351.42 4000.38 5351.42 4001.57 5351.42 4002.86 5351.42 4004.16 5351.42 4005.6 5351.42 4007.19 5351.42L4007.26 5334.99"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M4074.56 5386.62V5372.54H4059.05C4053.43 5372.54 4049.52 5371.43 4047.32 5369.2 4045.14 5367 4044.05 5363.03 4044.05 5357.29V5349.07H4069.86V5337.34H4044.05V5250.51H4027.63V5337.34H4011.2V5349.07H4027.63V5355.52C4027.63 5366.35 4030.14 5374.23 4035.18 5379.17 4040.24 5384.14 4048.26 5386.62 4059.23 5386.62H4074.56"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M4132.04 5299.79C4119.38 5299.79 4110.61 5298.34 4105.72 5295.43 4100.83 5292.52 4098.38 5287.54 4098.38 5280.5 4098.38 5274.91 4100.22 5270.46 4103.88 5267.16 4107.55 5263.88 4112.54 5262.25 4118.84 5262.25 4127.55 5262.25 4134.52 5265.32 4139.78 5271.48 4145.04 5277.67 4147.66 5285.88 4147.66 5296.13V5299.79H4132.04M4164.09 5305.99V5250.51H4147.66V5264.59C4144.05 5258.97 4139.54 5254.82 4134.13 5252.16 4128.73 5249.5 4122.12 5248.16 4114.3 5248.16 4104.4 5248.16 4096.53 5250.93 4090.68 5256.45 4084.87 5262 4081.96 5269.42 4081.96 5278.71 4081.96 5289.56 4085.61 5297.74 4092.92 5303.24 4100.23 5308.76 4111.14 5311.52 4125.66 5311.52H4147.66V5313.29C4147.66 5320.93 4145.26 5326.85 4140.44 5331.03 4135.65 5335.23 4128.91 5337.34 4120.24 5337.34 4114.71 5337.34 4109.34 5336.75 4104.11 5335.58 4098.88 5334.4 4093.84 5332.64 4089 5330.3V5344.38C4094.84 5346.72 4100.51 5348.48 4106.01 5349.66 4111.54 5350.83 4116.9 5351.42 4122.11 5351.42 4136.19 5351.42 4146.7 5347.65 4153.64 5340.13 4160.61 5332.62 4164.09 5321.24


4164.09 5305.99"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M4270.55 5344.38V5330.3C4265.95 5332.64 4261.35 5334.4 4256.73 5335.58 4252.13 5336.75 4247.49 5337.34 4242.79 5337.34 4232.26 5337.34 4224.08 5334.05 4218.26 5327.47 4212.45 5320.9 4209.54 5311.67 4209.54 5299.79 4209.54 5287.91 4212.45 5278.68 4218.26 5272.11 4224.08 5265.53 4232.26 5262.25 4242.79 5262.25 4247.49 5262.25 4252.13 5262.83 4256.73 5264 4261.35 5265.18 4265.95 5266.94 4270.55 5269.29V5255.2C4266.05 5252.86 4261.39 5251.1 4256.58 5249.93 4251.79 5248.75 4246.68 5248.16 4241.25 5248.16 4226.51 5248.16 4214.8 5252.81 4206.13 5262.1 4197.45 5271.41 4193.11 5283.98 4193.11 5299.79 4193.11 5315.82 4197.5 5328.44 4206.27 5337.63 4215.05 5346.82 4227.06 5351.42 4242.32 5351.42 4247.28 5351.42 4252.12 5350.83 4256.84 5349.66 4261.55 5348.48 4266.13 5346.72 4270.55 5344.38"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M4384.46 5304.78V5297.45H4309.36C4310.07 5285.98 4313.42 5277.24 4319.41 5271.23 4325.4 5265.24 4333.73 5262.25 4344.41 5262.25 4350.63 5262.25 4356.64 5263.03 4362.46 5264.59 4368.27 5266.16 4374.04 5268.5 4379.76 5271.63V5257.55C4373.99 5254.5 4368.08 5252.16 4362.02 5250.55 4355.98 5248.96 4349.86 5248.16 4343.64 5248.16 4328.05 5248.16 4315.7 5252.72 4306.57 5261.84 4297.48 5270.98 4292.94 5283.34 4292.94 5298.91 4292.94 5315.02 4297.3 5327.79 4306.03 5337.23 4314.75 5346.69 4326.52 5351.42 4341.34 5351.42 4354.63 5351.42 4365.14 5347.24 4372.87 5338.88 4380.59 5330.52 4384.46 5319.15 4384.46 5304.78M4368.03 5309.18C4367.91 5317.73 4365.4 5324.57 4360.51 5329.67 4355.65 5334.78 4349.21 5337.34 4341.19 5337.34 4332.1 5337.34 4324.81 5334.87 4319.34 5329.93 4313.88 5324.99 4310.74 5318.04 4309.91 5309.07L4368.03 5309.18"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M4509.07 5386.62C4501.13 5373.2 4495.23 5359.92 4491.37 5346.8 4487.53 5333.7 4485.61 5320.4 4485.61 5306.9 4485.61 5293.43 4487.55 5280.09 4491.44 5266.86 4495.33 5253.66 4501.2 5240.39 4509.07 5227.04H4494.85C4486.27 5240.76 4479.84 5254.23 4475.56 5267.45 4471.31 5280.7 4469.18 5293.85 4469.18 5306.9 4469.18 5319.91 4471.3 5333 4475.53 5346.17 4479.75 5359.38 4486.2 5372.86 4494.85 5386.62H4509.07"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M4558.75 5365.5V5264.59H4579.88C4597.72 5264.59 4610.79 5268.65 4619.07 5276.77 4627.36 5284.88 4631.5 5297.69 4631.5 5315.19 4631.5 5332.57 4627.36 5345.29 4619.07 5353.36 4610.79 5361.45 4597.72 5365.5 4579.88 5365.5H4558.75M4539.98 5379.58H4577.2C4602.38 5379.58 4620.84 5374.37 4632.6 5363.96 4644.38 5353.57 4650.27 5337.31 4650.27 5315.19 4650.27 5292.92 4644.36 5276.57 4632.53 5266.13 4620.7 5255.72 4602.25 5250.51 4577.2 5250.51H4539.98V5379.58"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M4679.45 5379.58H4707.1L4739.26 5292.61 4771.6 5379.58H4799.13V5250.51H4780.36V5363.99L4747.87 5276.32H4730.71L4698.23 5363.99V5250.51H4679.45V5379.58"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M4835.92 5379.58H4854.69V5250.51H4835.92V5379.58"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M4887.38 5386.62H4901.61C4910.18 5372.86 4916.6 5359.38 4920.86 5346.17 4925.13 5333 4927.27 5319.91 4927.27 5306.9 4927.27 5293.85 4925.13 5280.7 4920.86 5267.45 4916.6 5254.23 4910.18 5240.76 4901.61 5227.04H4887.38C4895.25 5240.39 4901.13 5253.66 4905.02 5266.86 4908.9 5280.09 4910.84 5293.43 4910.84 5306.9 4910.84 5320.4 4908.9 5333.7 4905.02 5346.8 4901.13 5359.92 4895.25 5373.2 4887.38 5386.62"/> <path transform="matrix(.1,0,0,-.1,0,796)" stroke-width="28.346" stroke-linecap="butt" stroke-miterlimit="10" stroke-linejoin="miter" fill="none" stroke="#000000" d="M3396.64 1225.42 2520.28 1128.9"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M2414.62 1117.27 2547.74 1203.22 2520.28 1128.9 2563.26 1062.34" fill-rule="evenodd"/> <path transform="matrix(.1,0,0,-.1,0,796)" stroke-width="28.346" stroke-linecap="butt" stroke-miterlimit="10" stroke-linejoin="miter" fill="none" stroke="#000000" d="M2414.62 1117.27 2547.74 1203.22 2520.28 1128.9 2563.26 1062.34Z"/> <path transform="matrix(.1,0,0,-.1,0,796)" stroke-width="28.346" stroke-linecap="butt" stroke-miterlimit="10" stroke-linejoin="miter" fill="none" stroke="#000000" d="M1532.74 2170.11 1483.57 1554.85"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M1475.11 1448.89 1415.76 1595.81 1483.57 1554.85 1557.04 1584.52" fill-rule="evenodd"/> <path transform="matrix(.1,0,0,-.1,0,796)" stroke-width="28.346" stroke-linecap="butt" stroke-miterlimit="10" stroke-linejoin="miter" fill="none" stroke="#000000" d="M1475.11 1448.89 1415.76 1595.81 1483.57 1554.85 1557.04 1584.52Z"/> <path transform="matrix(.1,0,0,-.1,0,796)" stroke-width="28.346" stroke-linecap="butt" stroke-miterlimit="10"

stroke-linejoin="miter" fill="none" stroke="#000000" d="M2443.28 2777.17 3324.41 1343"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M3380.05 1252.43 3245.48 1336.09 3324.41 1343 3366.24 1410.28" fill-rule="evenodd"/> <path transform="matrix(.1,0,0,-.1,0,796)" stroke-width="28.346" stroke-linecap="butt" stroke-miterlimit="10" stroke-linejoin="miter" fill="none" stroke="#000000" d="M3380.05 1252.43 3245.48 1336.09 3324.41 1343 3366.24 1410.28Z"/> <path transform="matrix(.1,0,0,-.1,0,796)" stroke-width="28.346" stroke-linecap="butt" stroke-miterlimit="10" stroke-linejoin="miter" fill="none" stroke="#000000" d="M4870.61 2257.28 4373.3 1346.54"/> <path transform="matrix(.1,0,0,-.1,0,796)" d="M4322.36 1253.24 4328.09 1411.59 4373.3 1346.54 4452.48 1343.67" fill-rule="evenodd"/> <path transform="matrix(.1,0,0,-.1,0,796)" stroke-width="28.346" stroke-linecap="butt" stroke-miterlimit="10" stroke-linejoin="miter" fill="none" stroke="#000000" d="M4322.36 1253.24 4328.09 1411.59 4373.3 1346.54 4452.48 1343.67Z"/> </svg>

The user interacts with the Debug Host (e.g. laptop), which is running a debugger (e.g. gdb). The debugger communicates with a Debug Translator (e.g. OpenOCD, which may include a hardware driver) to communicate with Debug Transport Hardware (e.g. Olimex USB-JTAG adapter). The Debug Transport Hardware connects the Debug Host to the hardware platform's Debug Transport Module (DTM). The DTM provides access to one or more Debug Modules (DMs) using the Debug Module Interface (DMI).

Each hart in the hardware platform is controlled by exactly one DM. Harts may be heterogeneous. There is no further limit on the hart-DM mapping, but usually all harts in a single core are controlled by the same DM. In most hardware platforms there will only be one DM that controls all the harts in the hardware platform.

DMs provide run control of their harts in the hardware platform. Abstract commands provide access to GPRs. Additional registers are accessible through abstract commands or by writing programs to the optional Program Buffer.

The Program Buffer allows the debugger to execute arbitrary instructions on a hart. This mechanism can also be used to access memory. An optional system bus access block allows memory accesses without using a RISC-V hart to perform the access.

Each RISC-V hart may implement a Trigger Module. When trigger conditions are met, harts will halt and inform the debug module that they have halted.

Chapter 3. Debug Module (DM) (non-ISA extension)

The Debug Module implements a translation interface between abstract debug operations and their specific implementation. It might support the following operations:

Give the debugger necessary information about the implementation. (Required)

Allow any individual hart to be halted and resumed. (Required)

Provide status on which harts are halted. (Required)

Provide abstract read and write access to a halted hart's GPRs. (Required)

Provide access to a reset signal that allows debugging from the very first instruction after reset. (Required)

Provide a mechanism to allow debugging harts immediately out of reset (regardless of the reset cause). (Optional)

Provide abstract access to non-GPR hart registers. (Optional)

Provide a Program Buffer to force the hart to execute arbitrary instructions. (Optional)

Allow multiple harts to be halted, resumed, and/or reset at the same time. (Optional)

Allow memory access from a hart's point of view. (Optional)

Allow direct System Bus Access. (Optional)

Group harts. When any hart in the group halts, they all halt. (Optional)

Respond to external triggers by halting each hart in a configured group. (Optional)

Signal an external trigger when a hart in a group halts. (Optional)

In order to be compatible with this specification an implementation must:

Implement all the required features listed above.

Implement at least one of Program Buffer, System Bus Access, or Abstract Access Memory command mechanisms.

Do at least one of:

Implement the Program Buffer.

Implement abstract access to all registers that are visible to software running on the hart including all the registers that are present on the hart and listed in Table #tab:regno.

Implement abstract access to at least all GPRs, , and , and advertise the implementation as conforming to the `Minimal RISC-V Debug Specification '', instead of the` RISC-V Debug Specification ".

A single DM can debug up to 2^{20} harts.

3.1. Debug Module Interface (DMI)

Debug Modules are subordinates on a bus called the Debug Module Interface (DMI). The bus manager is the Debug Transport Module(s). The Debug Module Interface can be a trivial bus with one manager and one subordinate (see [dmi_signals]), or use a more full-featured bus like TileLink or the AMBA Advanced Peripheral Bus. The details are left to the system designer.

The DMI uses between 7 and 32 address bits. Each address points at a single 32-bit register that can be read or written. The bottom of the address space is used for the first (and usually only) DM. Extra space can be used for custom debug devices, other cores, additional DMs, etc. If there are additional DMs on this DMI, the base address of the next DM in the DMI address space is given in .

The Debug Module is controlled via register accesses to its DMI address space.

3.2. Reset Control

There are two methods that allow a debugger to reset harts. resets all the harts in the hardware platform, as well as all other parts of the hardware platform except for the Debug Modules, Debug Transport Modules, and Debug Module Interface. Exactly what is affected by this reset is implementation dependent, but it must be possible to debug programs from the first instruction executed. resets all the currently selected harts. In this case an implementation may reset more harts than just the ones that are selected. The debugger can discover which other harts are reset (if any) by selecting them and checking and .

To perform either of these resets, the debugger first asserts the bit, and then clears it. The actual reset may start as soon as the bit is asserted, but may start an arbitrarily long time after the bit is deasserted. The reset itself may also take an arbitrarily long time. While the reset is on-going, harts are either in the running state, indicating it's possible to perform some abstract commands during this time, or in the unavailable state, indicating it's not possible to perform any abstract commands during this time. Once a hart's reset is complete, `havereset` becomes set. When a hart comes out of reset and or are set, the hart will immediately enter Debug Mode (halted state). Otherwise, if the hart was initially running it will execute normally (running state) and if the hart was initially halted it should now be running but may be halted.

There is no general, reliable way for the debugger to know when reset has actually begun.

The Debug Module's own state and registers should only be reset at power-up and while in is 0. If there is another mechanism to reset the DM, this mechanism must also reset all the harts accessible to the DM.

Due to clock and power domain crossing issues, it might not be possible to perform arbitrary DMI accesses across hardware platform reset. While or any external reset is asserted, the only supported DM operations are reading and writing . The behavior of other accesses is undefined.

When harts have been reset, they must set a sticky `havereset` state bit. The conceptual `havereset` state bits can be read for selected harts in and in . These bits must be set regardless of the cause of the reset. The `havereset` bits for the selected harts can be cleared by writing 1 to in . The `havereset` bits might or might not be cleared when is low.

3.3. Selecting Harts

Up to 2^{20} harts can be connected to a single DM. Commands issued to the DM only apply to the currently selected harts.

To enumerate all the harts, a debugger must first determine `HARTSELLEN` by writing all ones to (assuming the maximum size) and reading back the value to see which bits were actually set. Then it selects each hart starting from 0 until either in is 1, or the highest index (depending on `HARTSELLEN`) is reached.

The debugger can discover the mapping between hart indices and by using the interface to read , or by reading the hardware platform's configuration structure.

3.3.1. Selecting a Single Hart

All debug modules must support selecting a single hart. The debugger can select a hart by writing its index to . Hart indexes start at 0 and are contiguous until the final index.

3.3.2. Selecting Multiple Harts

Debug Modules may implement a Hart Array Mask register to allow selecting multiple harts at once. The nth bit in the Hart Array Mask register applies to the hart with index n. If the bit is 1 then the hart is selected. Usually a DM will have a Hart Array Mask register exactly wide enough to select all the harts it supports, but it's allowed to tie any of these bits to 0.

The debugger can set bits in the hart array mask register using and , then apply actions to all selected harts by setting . If this feature is supported, multiple harts can be halted, resumed, and reset simultaneously. The state of the hart array mask register is not affected by setting or clearing .

Execution of Abstract Commands ignores this mechanism and only applies to the hart selected by .

3.4. Hart DM States

Every hart that can be selected is in exactly one of the following four DM states: non-existent, unavailable, running, or halted. Which state the selected harts are in is reflected by , , , , , , , and .

Harts are nonexistent if they will never be part of this hardware platform, no matter how long a user waits. E.g. in a simple single-hart hardware platform only one hart exists, and all others are nonexistent. Debuggers may assume that a hardware platform has no harts with indexes higher than the first nonexistent one.

Harts are unavailable if they might exist/become available at a later time, or if there are other harts with higher indexes than this one. Harts may be unavailable for a variety of reasons including being reset, temporarily powered down, and not being plugged into the hardware platform. That means harts might become available or unavailable at any time, although these events should be rare in hardware platforms built to be easily debugged. There are no guarantees about the state of the hart when it becomes available.

Hardware platforms with very large number of harts may permanently disable some during manufacturing, leaving holes in the otherwise continuous hart index space. In order to let the debugger discover all harts, they must show up as unavailable even if there is no chance of them ever becoming available.

Harts are running when they are executing normally, as if no debugger was attached. This includes being in a low power mode or waiting for an interrupt, as long as a halt request will result in the hart being halted.

Harts are halted when they are in Debug Mode, only performing tasks on behalf of the debugger.

Which states a hart that is reset goes through is implementation dependent. Harts may be unavailable while reset is asserted, and some time after reset is deasserted. They might transition to running for some time after reset is deasserted. Finally they end up either running or halted, depending on and .

3.5. Run Control

For every hart, the Debug Module tracks 4 conceptual bits of state: halt request, resume ack, halt-on-reset request, and hart reset. (The hart reset and halt-on-reset request bits are optional.) These 4 bits reset to 0, except for resume ack, which may reset to either 0 or 1. The DM receives halted, running, and havereset signals from each hart. The debugger can observe the state of resume ack in and , and the state of halted, running, and havereset signals in , , , , , and . The state of the other bits cannot be observed directly.

When a debugger writes 1 to , each selected hart's halt request bit is set. When a running hart, or a hart just coming out of reset, sees its halt request bit high, it responds by halting, deasserting its running signal, and asserting its halted signal. Halted harts ignore their halt request bit.

When a debugger writes 1 to , each selected hart's resume ack bit is cleared and each selected, halted hart is sent a resume request. Harts respond by resuming, clearing their halted signal, and asserting their running signal. At the end of this process the resume ack bit is set. These status signals of all selected harts are reflected in , , , and . Resume requests are ignored by running harts.

When halt or resume is requested, a hart must respond in less than one second, unless it is unavailable. (How this is implemented is not further specified. A few clock cycles will be a more typical latency).

The DM can implement optional halt-on-reset bits for each hart, which it indicates by setting to 1. This means the DM implements the and bits. Writing 1 to sets the halt-on-reset request bit for each selected hart. When a hart's halt-on-reset request bit is set, the hart will immediately enter debug mode on the next deassertion of its reset. This is true regardless of the reset's cause. The hart's halt-on-reset request bit remains set until cleared by the debugger writing 1 to while the hart is selected, or by DM reset.

If the DM is reset while a hart is halted, it is whether that hart resumes. Debuggers should use to explicitly resume harts before clearing and disconnecting.

3.6. Halt Groups, Resume Groups, and External Triggers

An optional feature allows a debugger to place harts into two kinds of groups: halt groups and resume groups. It is also possible to add external triggers to a halt and resume groups. At any given time, each hart and each trigger is a member of exactly one halt group and exactly one resume group.

In both halt and resume groups, group 0 is special. Harts in group 0 halt/resume as if groups aren't implemented at all.

When any hart in a halt group halts:

That hart halts normally, with reflecting the original cause of the halt.

All the other harts in the halt group that are running will quickly halt. for those harts should be set to 6, but may be set to 3. Other harts in the halt group that are halted but have started the process of resuming must also quickly become halted, even if they do resume briefly.

Any external triggers in that group are notified.

Adding a hart to a halt group does not automatically halt that hart, even if other harts in the group are already halted.

When an external trigger that's a member of the halt group fires:

All the harts in the halt group that are running will quickly halt. for those harts should be set to 6, but may be set to 3. Other harts in the halt group that are halted but have started the process of resuming must also quickly become halted, even if they do resume briefly.

When any hart in a resume group resumes:

All the other harts in that group that are halted will quickly resume as soon as any currently executing abstract commands have completed. Each hart in the group sets its resume ack bit as soon as it has resumed. Harts that are in the process of halting should complete that process and stay halted.

Any external triggers in that group are notified.

Adding a hart to a resume group does not automatically resume that hart, even if other harts in the group are currently running.

When an external trigger that's a member of the resume group fires:

All the harts in that group that are halted will quickly resume as soon as any currently executing abstract commands have completed. Each hart in the group sets its resume ack bit as soon as it has resumed. Harts that are in the process of halting should complete that process and stay halted.

External triggers are abstract concepts that can signal the DM and/or receive signals from the DM. This configuration is done through , where external triggers are referred to by a number. Commonly, external triggers are capable of sending a signal from the hardware platform into the DM, as well as receiving a signal from the DM to take their own action on. It is also allowable for an external trigger to be input-only or output-only. By convention external triggers 0–7 are bidirectional, triggers 8–11 are input-only, and triggers 12–15 are output-only but this is not required.

External triggers could be used to implement near simultaneous halting/resuming of all cores in a hardware platform, when not all cores are RISC-V cores.

When the DM is reset, all harts must be placed in the lowest-numbered halt and resume groups that they can be in. (This will usually be group 0.)

Some designs may choose to hardcode hart groups to a group other than group 0, meaning it is never possible to halt or resume just a single hart. This is explicitly allowed. In that case it must be possible to discover the groups by using even if it's not possible to change the configuration.

3.7. Abstract Commands

The DM supports a set of abstract commands, most of which are optional. Depending on the implementation, the debugger may be able to perform some abstract commands even when the selected hart is not halted. Debuggers can only determine which abstract commands are supported by a given hart in a given state (running, halted, or held in reset) by attempting them and then looking at in to see if they were successful. Commands may be supported with some options set, but not with other options set. If a command has unsupported options set or if bits that are defined as 0 aren't 0, then the DM must set to 2 (not supported).

Example: Every DM must support the Access Register command, but might not support accessing CSRs. If the debugger requests to read a CSR in that case, the command will return `not supported."

Debuggers execute abstract commands by writing them to . They can determine whether an abstract command is complete by reading in . If the debugger starts a new command while is set, becomes 1 (busy), the currently executing command still gets to run to completion, but any error generated by the currently executing command is lost. After completion, indicates whether the command was successful or not. Commands may fail because a hart is not halted, not running, unavailable, or because they encounter an error during execution.

If the command takes arguments, the debugger must write them to the `data` registers before writing to . If a command returns results, the Debug Module must ensure they are placed in the `data` registers before is cleared. Which `data` registers are used for the arguments is described in Table #tab:datareg. In all cases the least-significant word is placed in the lowest-numbered `data` register. The argument width depends on the command being executed, and is DXLEN where not explicitly specified.

|r|l|l|l| Argument Width & arg0/return value & arg1 & arg2
& & `data1` & `data2`
& , `data1` & `data2`, `data3` & `data4`, `data5`
& –`data3` & `data4`–`data7` & `data8`–`data11`

The Abstract Command interface is designed to allow a debugger to write commands as fast as possible, and then later check whether they completed without error. In the common case the debugger will be much slower than the target and commands succeed, which allows for maximum throughput. If there is a failure, the interface ensures that no commands execute after the failing one. To discover which command failed, the debugger has to look at the state of the DM (e.g. contents of) or hart (e.g. contents of a register modified by a Program Buffer program) to determine which one failed.

Before starting an abstract command, a debugger must ensure that , , and are all 0.

While an abstract command is executing (in is high), a debugger must not change , and must not write 1 to , , , , or .

If an abstract command does not complete in the expected time and appears to be hung, the debugger can try to reset the hart (using or). If that doesn't clear , then it can try resetting the Debug Module (using).

If an abstract command is started while the selected hart is unavailable or if a hart becomes unavailable while executing an abstract command, then the Debug Module may terminate the abstract command, setting low, and to 4 (halt/resume). Alternatively, the command could just appear to be hung (never goes low).

3.7.1. Abstract Command Listing

This section describes each of the different abstract commands and how their fields should be interpreted when they are written to .

Each abstract command is a 32-bit value. The top 8 bits contain which determines the kind of command. Table #tab:cmdtype lists all commands.

|r|l|l|l| & Command & Page
& Access Register Command &
& Quick Access &
& Access Memory Command &

3.8. Program Buffer

To support executing arbitrary instructions on a halted hart, a Debug Module can include a Program Buffer that a debugger can write small programs to. DMs that support all necessary functionality using abstract commands only may choose to omit the Program Buffer.

A debugger can write a small program to the Program Buffer, and then execute it exactly once with the Access Register Abstract Command, setting the bit in . The debugger can write whatever program it likes (including jumps out of the Program Buffer), but the program must end with `ebreak` or `c.ebreak`. An implementation may support an implicit `ebreak` that is executed when a hart runs off the end of the Program Buffer. This is indicated by . With this feature, a Program Buffer of just 2 32-bit words can offer efficient debugging.

While these programs are executed, the hart does not leave Debug Mode (see Section [debugmode]). If an exception is encountered during execution of the Program Buffer, no more instructions are executed, the hart remains in Debug Mode, and is set to 3 (`exception error`). If the debugger executes a program that doesn't terminate with an `ebreak` instruction, the hart will remain in Debug Mode and the debugger will lose control of the hart.

If is 1 then the following apply:

must be 1.

If the debugger writes a compressed instruction into the Program Buffer, it must be placed into the lower 16 bits and accompanied by a compressed `nop` in the upper 16 bits.

This requirement on the debugger for the case of equal to 1 is to accommodate hardware designs that prefer to stuff instructions directly into the pipeline when halted, instead of having the Program Buffer exist in the address space somewhere.

The Program Buffer may be implemented as RAM which is accessible to the hart. A debugger can determine if this is the case by executing small programs that attempt to write and read back relative to while executing from the Program Buffer. If so, the debugger has more flexibility in what it can do with the program buffer.

3.9. Overview of Hart Debug States

Figure #fig:abstract_sm[1.1] shows a conceptual view of the states passed through by a hart during run/halt debugging as influenced by the different fields of , , , and .

[fig/abstract_commands] | *fig/abstract_commands.pdf*

Figure 1. Run/Halt Debug State Machine for single-hart hardware platforms. As only a small amount of state is visible to the debugger, the states and transitions are conceptual.

3.10. System Bus Access

A debugger can access memory from a hart's point of view using a Program Buffer or the Abstract Access Memory command. (Both these features are optional.) A Debug Module may also include a System Bus Access block to provide memory access without involving a hart, regardless of whether Program Buffer is implemented. The System Bus Access block uses physical addresses.

The System Bus Access block may support 8-, 16-, 32-, 64-, and 128-bit accesses. Table #tab:sbdatabits shows which bits in `sbdata` are used for each access size.

|r|l| Access Size & Data Bits
& bits 7:0
& bits 15:0
&
& ,
& , , ,

Depending on the microarchitecture, data accessed through System Bus Access might not always be coherent with that observed by each hart. It is up to the debugger to enforce coherency if the implementation does not. This specification does not define a standard way to do this. Possibilities may include writing to special memory-mapped locations, or executing special instructions via the Program Buffer.

Implementing a System Bus Access block has several benefits even when a Debug Module also implements a Program Buffer. First, it is possible to access memory in a running system with minimal impact. Second, it may improve performance when accessing memory. Third, it may provide access to devices that a hart does not have access to.

3.11. Minimally Intrusive Debugging

Depending on the task it is performing, some harts can only be halted very briefly. There are several mechanisms that allow accessing resources in such a running system with a minimal impact on the running hart.

First, an implementation may allow some abstract commands to execute without halting the hart.

Second, the Quick Access abstract command can be used to halt a hart, quickly execute the contents of the Program Buffer, and let the hart run again. Combined with instructions that allow Program Buffer code to access the `data` registers, as described in , this can be used to quickly perform a memory or register access. For some hardware platforms this will be too intrusive, but many hardware platforms that can't be halted can bear an occasional hiccup of a hundred or less cycles.

Third, if the System Bus Access block is implemented, it can be used while a hart is running to access system memory.

3.12. Security

To protect intellectual property it may be desirable to lock access to the Debug Module. To allow access during a manufacturing process and not afterwards, a reasonable solution could be to add a fuse bit to the Debug Module that can be used to be permanently disable it. Since this is technology specific, it is not further addressed in this spec.

Another option is to allow the DM to be unlocked only by users who have an access key. Between , , and arbitrarily complex authentication mechanism can be supported. When is clear, the DM must not interact with the rest of the hardware platform, nor expose details about the harts connected to the DM. All DM registers should read 0, while writes should be ignored, with the following mandatory exceptions:

in is readable.

in is readable.

in is readable.

in is readable and writable.

is readable and writable.

Implementations where it's not possible to unlock the DM by using should not implement that register.

3.13. Version Detection

To detect the version of the Debug Module with a minimum of side effects, use the following procedure:

Read .

If is 0 or is 1:

1. Write , preserving , , , and from the value that was read, setting , and clearing all the other bits.
2. Read until is high.

Read , which contains .

If it was necessary to clear , this might have the following unavoidable side effects:

is cleared, potentially preventing a halt request made by a previous debugger from taking effect.

is cleared, potentially preventing a resume request made by a previous debugger from taking effect.

is deasserted, releasing the hardware platform from reset if a previous debugger had set it.

is asserted, releasing the DM from reset. This in itself is not observable by any harts.

This procedure is guaranteed to work in future versions of this spec. The meaning of the bits where , , , and currently reside might change, but preserving them will have no side effects. Clearing the bits of not explicitly mentioned here will have no side effects beyond the ones mentioned above.

3.14. Debug Module Registers

The registers described in this section are accessed over the DMI bus. Each DM has a base address (which is 0 for the first DM). The register addresses below are offsets from this base address.

Debug Module DMI Registers that are unimplemented or not mentioned in the table below return 0 when read. Writing them has no effect.

Address	Name	Page
Continued on next page		
0x04	Abstract Data 0 (`data0`)	
0x05	Abstract Data 1 (`data1`)	
0x06	Abstract Data 2 (`data2`)	
0x07	Abstract Data 3 (`data3`)	
0x08	Abstract Data 4 (`data4`)	
0x09	Abstract Data 5 (`data5`)	
0x0a	Abstract Data 6 (`data6`)	
0x0b	Abstract Data 7 (`data7`)	
0x0c	Abstract Data 8 (`data8`)	
0x0d	Abstract Data 9 (`data9`)	
0x0e	Abstract Data 10 (`data10`)	
0x0f	Abstract Data 11 (`data11`)	
0x10	Debug Module Control (`dmcontrol`)	
0x11	Debug Module Status (`dmstatus`)	
0x12	Hart Info (`hartinfo`)	
0x13	Halt Summary 1 (`haltsum1`)	
0x14	Hart Array Window Select (`hawindowsel`)	
0x15	Hart Array Window (`hawindow`)	
0x16	Abstract Control and Status (`abstractcs`)	
0x17	Abstract Command (`command`)	
0x18	Abstract Command Autoexec (`abstractauto`)	
0x19	Configuration Structure Pointer 0 (`confstrptr0`)	
0x1a	Configuration Structure Pointer 1 (`confstrptr1`)	
0x1b	Configuration Structure Pointer 2 (`confstrptr2`)	
0x1c	Configuration Structure Pointer 3 (`confstrptr3`)	

Address	Name	Page
0x1d	Next Debug Module (`nextdm`)	
0x1f	Custom Features (`custom`)	
0x20	Program Buffer 0 (`progbuf0`)	
0x21	Program Buffer 1 (`progbuf1`)	
0x22	Program Buffer 2 (`progbuf2`)	
0x23	Program Buffer 3 (`progbuf3`)	
0x24	Program Buffer 4 (`progbuf4`)	
0x25	Program Buffer 5 (`progbuf5`)	
0x26	Program Buffer 6 (`progbuf6`)	
0x27	Program Buffer 7 (`progbuf7`)	
0x28	Program Buffer 8 (`progbuf8`)	
0x29	Program Buffer 9 (`progbuf9`)	
0x2a	Program Buffer 10 (`progbuf10`)	
0x2b	Program Buffer 11 (`progbuf11`)	
0x2c	Program Buffer 12 (`progbuf12`)	
0x2d	Program Buffer 13 (`progbuf13`)	
0x2e	Program Buffer 14 (`progbuf14`)	
0x2f	Program Buffer 15 (`progbuf15`)	
0x30	Authentication Data (`authdata`)	
0x32	Debug Module Control and Status 2 (`dmcs2`)	
0x34	Halt Summary 2 (`haltsum2`)	
0x35	Halt Summary 3 (`haltsum3`)	
0x37	System Bus Address 127:96 (`sbaddress3`)	
0x38	System Bus Access Control and Status (`sbcs`)	
0x39	System Bus Address 31:0 (`sbaddress0`)	
0x3a	System Bus Address 63:32 (`sbaddress1`)	
0x3b	System Bus Address 95:64 (`sbaddress2`)	
0x3c	System Bus Data 31:0 (`sbdata0`)	
0x3d	System Bus Data 63:32 (`sbdata1`)	
0x3e	System Bus Data 95:64 (`sbdata2`)	
0x3f	System Bus Data 127:96 (`sbdata3`)	
0x40	Halt Summary 0 (`haltsum0`)	
0x70	Custom Features 0 (`custom0`)	

Address	Name	Page
0x71	Custom Features 1 (`custom1`)	
0x72	Custom Features 2 (`custom2`)	
0x73	Custom Features 3 (`custom3`)	
0x74	Custom Features 4 (`custom4`)	
0x75	Custom Features 5 (`custom5`)	
0x76	Custom Features 6 (`custom6`)	
0x77	Custom Features 7 (`custom7`)	
0x78	Custom Features 8 (`custom8`)	
0x79	Custom Features 9 (`custom9`)	
0x7a	Custom Features 10 (`custom10`)	
0x7b	Custom Features 11 (`custom11`)	
0x7c	Custom Features 12 (`custom12`)	
0x7d	Custom Features 13 (`custom13`)	
0x7e	Custom Features 14 (`custom14`)	
0x7f	Custom Features 15 (`custom15`)	

Table 2. Debug Module Debug Bus Registers

3.14.1. Debug Module Status (`dmstatus`, at 0x11)

[dmDmstatus]# This register reports status for the overall Debug Module as well as the currently selected harts, as defined in . Its address will not change in the future, because it contains .

This entire register is read-only.

31	25	24		23		22		21	20	19	
latexmath:[$	0	$]		latexmath:[$	ndmresetpending	$]		latexmath:[$	stickyunavail	$]	
latexmath:[$	impebreak	$]		latexmath:[$	0	$]		latexmath:[$	allhavereset	$]	
7		1		1		1		2		1	

18		17		16		15		14		13	
latexmath:[$	anyhavereset	$]		latexmath:[$	allresumeack	$]		latexmath:[$	anyresumeack	$]	
latexmath:[$	allnonexistent	$]		latexmath:[$	anynonexistent	$]		latexmath:[$	allunavail	$]	
1		1		1		1		1		1	

12		11		10		9		8		7	
latexmath:[$	anyunavail	$]		latexmath:[$	allrunning	$]		latexmath:[$	anyrunning	$]	

12		11		10		9		8		7	
latexmath:[$	allhalted	$]		latexmath:[$	anyhalted	$]		latexmath:[$	authenticated	$]	
1		1		1		1		1		1	

6		5		4		3	0
latexmath:[$	authbusy	$]		latexmath:[$	hasresethaltreq	$]	
latexmath:[$	confstrptrvalid	$]		latexmath:[$	version	$]	
1		1		1		4	

Field	Description	Access	Reset
Continued on next page			
[dmDmstatusNdmresetpending]# |ndmresetpending|	0 (false): Unimplemented, or is zero and no ndmreset is currently in progress. 1 (true): is currently nonzero, or there is an ndmreset in progress.	R	-
[dmDmstatusStickyunavail]# |stickyunavail|	0 (current): The per-hart `unavail` bits reflect the current state of the hart. 1 (sticky): The per-hart `unavail` bits are sticky. Once they are set, they will not clear until the debugger acknowledges them using .	R	Preset
[dmDmstatusImpebreak]# |impebreak|	If 1, then there is an implicit `ebreak` instruction at the non-existent word immediately after the Program Buffer. This saves the debugger from having to write the `ebreak` itself, and allows the Program Buffer to be one word smaller. This must be 1 when is 1.	R	Preset
[dmDmstatusAllhavereset]# |allhavereset|	This field is 1 when all currently selected harts have been reset and reset has not been acknowledged for any of them.	R	-
[dmDmstatusAnyhavereset]# |anyhavereset|	This field is 1 when at least one currently selected hart has been reset and reset has not been acknowledged for that hart.	R	-

Field	Description	Access	Reset
[dmDmstatusAllresumeack]# |allresumeack|	This field is 1 when all currently selected harts have their resume ack bit set.	R	-
[dmDmstatusAnyresumeack]# |anyresumeack|	This field is 1 when any currently selected hart has its resume ack bit set.	R	-
[dmDmstatusAllnonexistent]# |allnonexistent|	This field is 1 when all currently selected harts do not exist in this hardware platform.	R	-
[dmDmstatusAnynonexistent]# |anynonexistent|	This field is 1 when any currently selected hart does not exist in this hardware platform.	R	-
[dmDmstatusAllunavail]# |allunavail|	This field is 1 when all currently selected harts are unavailable, or (if is 1) were unavailable without that being acknowledged.	R	-
[dmDmstatusAnyunavail]# |anyunavail|	This field is 1 when any currently selected hart is unavailable, or (if is 1) was unavailable without that being acknowledged.	R	-
[dmDmstatusAllrunning]# |allrunning|	This field is 1 when all currently selected harts are running.	R	-
[dmDmstatusAnyrunning]# |anyrunning|	This field is 1 when any currently selected hart is running.	R	-
[dmDmstatusAllhalted]# |allhalted|	This field is 1 when all currently selected harts are halted.	R	-
[dmDmstatusAnyhalted]# |anyhalted|	This field is 1 when any currently selected hart is halted.	R	-
[dmDmstatusAuthenticated]# |authenticated|	0 (false): Authentication is required before using the DM. 1 (true): The authentication check has passed. On components that don't implement authentication, this bit must be preset as 1.	R	Preset

Field	Description	Access	Reset
[dmDmstatusAuthbusy]# |authbusy|	0 (ready): The authentication module is ready to process the next read/write to . 1 (busy): The authentication module is busy. Accessing results in unspecified behavior. only becomes set in immediate response to an access to .	R	0
[dmDmstatusHasresethaltreq]# |hasresethaltreq|	1 if this Debug Module supports halt-on-reset functionality controllable by the and bits. 0 otherwise.	R	Preset
[dmDmstatusConfstrptrvalid]# |confstrptrvalid|	0 (invalid): –hold information which is not relevant to the configuration structure. 1 (valid): –hold the address of the configuration structure.	R	Preset
|version|	0 (none): There is no Debug Module present. 1 (0.11): There is a Debug Module and it conforms to version 0.11 of this specification. 2 (0.13): There is a Debug Module and it conforms to version 0.13 of this specification. 3 (1.0): There is a Debug Module and it conforms to version 1.0 of this specification. 15 (custom): There is a Debug Module but it does not conform to any available version of this spec.	R	3

3.14.2. Debug Module Control (`dmcontrol`, at 0x10)

[dmDmcontrol]# This register controls the overall Debug Module as well as the currently selected harts, as defined in .

[hartsel]# Throughout this document we refer to , which is combined with . While the spec allows for

20 bits, an implementation may choose to implement fewer than that. The actual width of is called HARTSELLEN. It must be at least 0 and at most 20. A debugger should discover HARTSELLEN by writing all ones to (assuming the maximum size) and reading back the value to see which bits were actually set. Debuggers must not change while an abstract command is executing.

There are separate and bits so that it is possible to write without changing the halt-on-reset request bit for each selected hart, when not all selected harts have the same configuration.

On any given write, a debugger may only write 1 to at most one of the following bits: , , , , and . The others must be written 0.

[resethaltreq]# is an optional internal bit of per-hart state that cannot be read, but can be written with and .

[keepalive]# is an optional internal bit of per-hart state. When it is set, it suggests that the hardware should attempt to keep the hart available for the debugger, e.g. by keeping it from entering a low-power state once powered on. Even if the bit is implemented, hardware might not be able to keep a hart available. The bit is written through and .

For forward compatibility, will always be readable when bit 1 () is 0 and bit 0 () is 1.

31		30		29		28		27	
latexmath:[$	haltreq	$]		latexmath:[$	resumereq	$]		latexmath:[$	hartreset
$]		latexmath:[$	ackhavereset	$]		latexmath:[$	ackunavail	$]	
1		1		1		1		1	

26		25	16	15	6	5		4	
latexmath:[$	hasel	$]		latexmath:[$	hartsello	$]		latexmath:[$	hartselhi
$]		latexmath:[$	setkeepalive	$]		latexmath:[$	clrkeepalive	$]	
1		10		10		1		1	

3		2		1		0	
latexmath:[$	setresethaltreq	$]		latexmath:[$	clrresethaltreq	$]	
latexmath:[$	ndmreset	$]		latexmath:[$	dmactive	$]	
1		1		1		1	

Field	Description	Access	Reset
Continued on next page			

Field	Description	Access	Reset
[dmDmcontrolHaltreq]# |haltreq|	Writing 0 clears the halt request bit for all currently selected harts. This may cancel outstanding halt requests for those harts. Writing 1 sets the halt request bit for all currently selected harts. Running harts will halt whenever their halt request bit is set. Writes apply to the new value of and .	WARZ	-
[dmDmcontrolResumereq]# |resumereq|	Writing 1 causes the currently selected harts to resume once, if they are halted when the write occurs. It also clears the resume ack bit for those harts. is ignored if is set. Writes apply to the new value of and .	W1	-
[dmDmcontrolHartreset]# |hartreset|	This optional field writes the reset bit for all the currently selected harts. To perform a reset the debugger writes 1, and then writes 0 to deassert the reset signal. While this bit is 1, the debugger must not change which harts are selected. If this feature is not implemented, the bit always stays 0, so after writing 1 the debugger can read the register back to see if the feature is supported. Writes apply to the new value of and .	WARL	0
[dmDmcontrolAckhavereset]# |ackhavereset|	0 (nop): No effect. 1 (ack): Clears `havereset` for any selected harts. Writes apply to the new value of and .	W1	-

Field	Description	Access	Reset
[dmDmcontrolAckunavail]# |ackunavail|	0 (nop): No effect. 1 (ack): Clears `unavail` for any selected harts that are currently available. Writes apply to the new value of and .	W1	-
[dmDmcontrolHasel]# |hasel|	Selects the definition of currently selected harts. 0 (single): There is a single currently selected hart, that is selected by . 1 (multiple): There may be multiple currently selected harts – the hart selected by , plus those selected by the hart array mask register. An implementation which does not implement the hart array mask register must tie this field to 0. A debugger which wishes to use the hart array mask register feature should set this bit and read back to see if the functionality is supported.	WARL	0
[dmDmcontrolHartsello]# |hartsello|	The low 10 bits of : the DM-specific index of the hart to select. This hart is always part of the currently selected harts.	WARL	0
[dmDmcontrolHartselhi]# |hartselhi|	The high 10 bits of : the DM-specific index of the hart to select. This hart is always part of the currently selected harts.	WARL	0
[dmDmcontrolSetkeepalive]# |setkeepalive|	This optional field sets for all currently selected harts, unless is simultaneously set to 1. Writes apply to the new value of and .	W1	-
[dmDmcontrolClrkeepalive]# |clrkeepalive|	This optional field clears for all currently selected harts. Writes apply to the new value of and .	W1	-

Field	Description	Access	Reset
[dmDmcontrolSetresethaltreq]# |setresethaltreq|	This optional field writes the halt-on-reset request bit for all currently selected harts, unless is simultaneously set to 1. When set to 1, each selected hart will halt upon the next deassertion of its reset. The halt-on-reset request bit is not automatically cleared. The debugger must write to to clear it. Writes apply to the new value of and . If is 0, this field is not implemented.	W1	-
[dmDmcontrolClrresethaltreq]# |clrresethaltreq|	This optional field clears the halt-on-reset request bit for all currently selected harts. Writes apply to the new value of and .	W1	-
[dmDmcontrolNdmreset]# |ndmreset|	This bit controls the reset signal from the DM to the rest of the hardware platform. The signal should reset every part of the hardware platform, including every hart, except for the DM and any logic required to access the DM. To perform a hardware platform reset the debugger writes 1, and then writes 0 to deassert the reset.	R/W	0

Field	Description	Access	Reset
|dmactive|	This bit serves as a reset signal for the Debug Module itself. After changing the value of this bit, the debugger must poll until has taken the requested value before performing any action that assumes the requested state change has completed. Hardware may take an arbitrarily long time to complete activation or deactivation and will indicate completion by setting to the requested value. 0 (inactive): The module's state, including authentication mechanism, takes its reset values (the bit is the only bit which can be written to something other than its reset value). Any accesses to the module may fail. Specifically, might not return correct data. 1 (active): The module functions normally. No other mechanism should exist that may result in resetting the Debug Module after power up. To place the Debug Module into a known state, a debugger may write 0 to , poll until is observed 0, write 1 to , and poll until is observed 1. Implementations may pay attention to this bit to further aid debugging, for example by preventing the Debug Module from being power gated while debugging is active.	R/W	0

3.14.3. Hart Info (`hartinfo`, at 0x12)

[dmHartinfo]# This register gives information about the hart currently selected by .

This register is optional. If it is not present it should read all-zero.

If this register is included, the debugger can do more with the Program Buffer by writing programs which explicitly access the `data` and/or `dscratch` registers.

This entire register is read-only.

31	24	23	20	19	17	16		15	12	11	0
latexmath:[$	0	$]		latexmath:[$	nscratch	$]		latexmath:[$	0	$]	
latexmath:[$	dataaccess	$]		latexmath:[$	datasize	$]		latexmath:[$	dataaddr	$]	
8		4		3		1		4		12	

Field	Description	Access	Reset
Continued on next page			
[dmHartinfoNscratch]# |nscratch|	Number of `dscratch` registers available for the debugger to use during program buffer execution, starting from . The debugger can make no assumptions about the contents of these registers between commands.	R	Preset
[dmHartinfoDataaccess]# |dataaccess|	0 (csr): The `data` registers are shadowed in the hart by CSRs. Each CSR is DXLEN bits in size, and corresponds to a single argument, per Table #tab:datareg. 1 (memory): The `data` registers are shadowed in the hart's memory map. Each register takes up 4 bytes in the memory map.	R	Preset
[dmHartinfoDatasize]# |datasize|	If is 0: Number of CSRs dedicated to shadowing the `data` registers. If is 1: Number of 32-bit words in the memory map dedicated to shadowing the `data` registers. If this value is non-zero, then the tt data registers must be traditional registers and not MRs. Since there are at most 12 `data` registers, the value in this register must be 12 or smaller.	R	Preset

Field	Description	Access	Reset
|dataaddr|	If is 0: The number of the first CSR dedicated to shadowing the `data` registers. If is 1: Address of RAM where the data registers are shadowed. This address is sign extended giving a range of -2048 to 2047, easily addressed with a load or store using as the address register.	R	Preset

3.14.4. Hart Array Window Select (`hawindowsel`, at 0x14)

[dmHawindowsel]# This register selects which of the 32-bit portion of the hart array mask register (see Section 1.3.2) is accessible in .

31	15	14	0
latexmath:[$	0	$]	
latexmath:[$	hawindowsel	$]	
17		15	

Field	Description	Access	Reset
Continued on next page			
|hawindowsel|	The high bits of this field may be tied to 0, depending on how large the array mask register is. E.g. on a hardware platform with 48 harts only bit 0 of this field may actually be writable.	WARL	0

3.14.5. Hart Array Window (`hawindow`, at 0x15)

[dmHawindow]# This register provides R/W access to a 32-bit portion of the hart array mask register (see Section 1.3.2). The position of the window is determined by . I.e. bit 0 refers to hart RdmHawindowsel * 32, while bit 31 refers to hart RdmHawindowsel * 32+31.

Since some bits in the hart array mask register may be constant 0, some bits in this register may be constant 0, depending on the current value of .

31	0
latexmath:[$	maskdata
$]	
32	

3.14.6. Abstract Control and Status (`abstractcs`, at 0x16)

[dmAbstractcs]# Writing this register while an abstract command is executing causes to become 1 (busy) once the command completes (busy becomes 0).

must be at least 1 to support RV32 harts, 2 to support RV64 harts, or 4 to support RV128 harts.

31	29	28	24	23	13	12	
latexmath:[$	0	$]		latexmath:[$	progbufsize	$]	
latexmath:[$	0	$]		latexmath:[$	busy	$]	
3		5		11		1	

11		10	8	7	4	3	0
latexmath:[$	relaxedpriv	$]		latexmath:[$	cmderr	$]	
latexmath:[$	0	$]		latexmath:[$	datacount	$]	
1		3		4		4	

Field	Description	Access	Reset
Continued on next page			
[dmAbstractcsProgbufsize]# |progbufsize|	Size of the Program Buffer, in 32-bit words. Valid sizes are 0 - 16.	R	Preset
[dmAbstractcsBusy]# |busy|	0 (ready): There is no abstract command currently being executed. 1 (busy): An abstract command is currently being executed. This bit is set as soon as is written, and is not cleared until that command has completed.	R	0

Field	Description	Access	Reset
[dmAbstractcsRelaxedpriv] # |relaxedpriv|	This optional bit controls whether program buffer and abstract memory accesses are performed with the exact and full set of permission checks that apply based on the current architectural state of the hart performing the access, or with a relaxed set of permission checks (e.g. PMP restrictions are ignored). The details of the latter are implementation-specific. 0 (full checks): Full permission checks apply. 1 (relaxed checks): Relaxed permission checks apply.	WARL	Preset

Field	Description	Access	Reset
[dmAbstractcsCmderr]# |cmderr|	Gets set if an abstract command fails. The bits in this field remain set until they are cleared by writing 1 to them. No abstract command is started until the value is reset to 0. This field only contains a valid value if is 0. 0 (none): No error. 1 (busy): An abstract command was executing while , , or was written, or when one of the `data` or `progbuf` registers was read or written. This status is only written if contains 0. 2 (not supported): The command in is not supported. It may be supported with different options set, but it will not be supported at a later time when the hart or system state are different. 3 (exception): An exception occurred while executing the command (e.g. while executing the Program Buffer). 4 (halt/resume): The abstract command couldn't execute because the hart wasn't in the required state (running/halted), or unavailable. 5 (bus): The abstract command failed due to a bus error (e.g. alignment, access size, or timeout). 6 (reserved): Reserved for future use. 7 (other): The command failed for another reason.	R/W1C	0

Field	Description	Access	Reset
|datacount|	Number of `data` registers that are implemented as part of the abstract command interface. Valid sizes are 1 – 12.	R	Preset

3.14.7. Abstract Command (`command`, at 0x17)

[dmCommand]# Writes to this register cause the corresponding abstract command to be executed.

Writing this register while an abstract command is executing causes to become 1 (busy) once the command completes (busy becomes 0).

If is non-zero, writes to this register are ignored.

inhibits starting a new command to accommodate debuggers that, for performance reasons, send several commands to be executed in a row without checking in between. They can safely do so and check at the end without worrying that one command failed but then a later command (which might have depended on the previous one succeeding) passed.

31	24	23	0
latexmath:[$	cmdtype	$]	
latexmath:[$	control	$]	
8		24	

Field	Description	Access	Reset
Continued on next page			
[dmCommandCmdtype]# |cmdtype|	The type determines the overall functionality of this abstract command.	WARZ	0
|control|	This field is interpreted in a command-specific manner, described for each abstract command.	WARZ	0

3.14.8. Abstract Command Autoexec (`abstractauto`, at 0x18)

[dmAbstractauto]# This register is optional. Including it allows more efficient burst accesses. A debugger can detect whether it is supported by setting bits and reading them back.

If this register is implemented then bits corresponding to implemented progbuf and data registers must be writable. Other bits must be hard-wired to 0.

If this register is written while an abstract command is executing then the write is ignored and becomes 1 (busy) once the command completes (busy becomes 0).

31	16	15	12	11	0
latexmath:[$	autoexecprogbuf	$]		latexmath:[$	0
$]		latexmath:[$	autoexecdata	$]	

31	16	15	12	11	0
16		4		12	

Field	Description	Access	Reset
Continued on next page			
[dmAbstractautoAutoexecprogbuf]# |autoexecprogbuf|	When a bit in this field is 1, read or write accesses to the corresponding `progbuf` word cause the DM to act as if the current value in was written there again after the access to `progbuf` completes.	WARL	0
|autoexecdata|	When a bit in this field is 1, read or write accesses to the corresponding `data` word cause the DM to act as if the current value in was written there again after the access to `data` completes.	WARL	0

3.14.9. Configuration Structure Pointer 0 (`confstrptr0`, at 0x19)

[dmConfstrptrZero]# When is set, reading this register returns bits 31:0 of the configuration structure pointer. Reading the other `confstrptr` registers returns the upper bits of the address.

When system bus access is implemented, this must be an address that can be used with the System Bus Access module. Otherwise, this must be an address that can be used to access the configuration structure from the hart with ID 0.

If is 0, then the `confstrptr` registers hold identifier information which is not further specified in this document.

The configuration structure itself is a data structure of the same format as the data structure pointed to by mconfigptr as described in the Privileged Spec.

This entire register is read-only.

31	0
latexmath:[$	addr
$]	
32	

3.14.10. Configuration Structure Pointer 1 (`confstrptr1`, at 0x1a)

[dmConfstrptrOne]# When is set, reading this register returns bits 63:32 of the configuration structure pointer. See for more details.

This entire register is read-only.

31	0
latexmath:[$	addr
$]	
32	

3.14.11. Configuration Structure Pointer 2 (`confstrptr2`, at 0x1b)

[dmConfstrptrTwo]# When is set, reading this register returns bits 95:64 of the configuration structure pointer. See for more details.

This entire register is read-only.

31	0
latexmath:[$	addr
$]	
32	

3.14.12. Configuration Structure Pointer 3 (`confstrptr3`, at 0x1c)

[dmConfstrptrThree]# When is set, reading this register returns bits 127:96 of the configuration structure pointer. See for more details.

This entire register is read-only.

31	0
latexmath:[$	addr
$]	
32	

3.14.13. Next Debug Module (`nextdm`, at 0x1d)

[dmNextdm]# If there is more than one DM accessible on this DMI, this register contains the base address of the next one in the chain, or 0 if this is the last one in the chain.

This entire register is read-only.

31	0
latexmath:[$	addr
$]	
32	

3.14.14. Abstract Data 0 (`data0`, at 0x04)

[dmDataZero]# through may be Message Registers, whose behavior is described in Section #sec:mr. These registers may be read or changed by abstract commands. indicates how many of them are implemented, starting at , counting up. Table #tab:datareg shows how abstract commands use these registers.

Accessing these registers while an abstract command is executing causes to be set to 1 (busy) if it is 0.

Attempts to write them while is set does not change their value.

The values in these registers might not be preserved after an abstract command is executed. The only guarantees on their contents are the ones offered by the command in question. If the command fails, no assumptions can be made about the contents of these registers.

31	0
latexmath:[$	data
$]	
32	

3.14.15. Program Buffer 0 (`progbuf0`, at 0x20)

[dmProgbufZero]# through must provide write access to the optional program buffer. It may also be possible for the debugger to read from the program buffer through these registers. If reading is not supported, then all reads return 0.

indicates how many `progbuf` registers are implemented starting at , counting up.

Accessing these registers while an abstract command is executing causes to be set to 1 (busy) if it is 0.

Attempts to write them while is set does not change their value.

31	0
latexmath:[$	data
$]	
32	

3.14.16. Authentication Data (`authdata`, at 0x30)

[dmAuthdata]# This register serves as a 32-bit serial port to/from the authentication module.

When is clear, the debugger can communicate with the authentication module by reading or writing this register. There is no separate mechanism to signal overflow/underflow.

31	0
latexmath:[$	data
$]	
32	

3.14.17. Debug Module Control and Status 2 (`dmcs2`, at 0x32)

[dmDmcsTwo]# This register contains DM control and status bits that didn't easily fit in and . All are optional.

If halt groups are not implemented, then will always be 0 when is 0.

If resume groups are not implemented, then will remain 0 even after 1 is written there.

The DM external triggers available to add to halt groups may be the same as or distinct from the DM external triggers available to add to resume groups.

31	12	11		10	7	6	2	1		0	
latexmath:[$	0	$]		latexmath:[$	grouptype	$]		latexmath:[$	dmexttrigger	$]	
latexmath:[$	group	$]		latexmath:[$	hgwrite	$]		latexmath:[$	hgselect	$]	
20		1		4		5		1		1	

Field	Description	Access	Reset
Continued on next page			
[dmDmcsTwoGrouptype]# |grouptype|	0 (halt): The remaining fields in this register configure halt groups. 1 (resume): The remaining fields in this register configure resume groups.	WARL	0
[dmDmcsTwoDmexttrigger]# |dmexttrigger|	This field contains the currently selected DM external trigger. If a non-existent trigger value is written here, the hardware will change it to a valid one or 0 if no DM external triggers exist.	WARL	0
[dmDmcsTwoGroup]# |group|	When is 0, contains the group of the hart specified by . When is 1, contains the group of the DM external trigger selected by . The value written to this field is ignored unless is also written 1. Group numbers are contiguous starting at 0, with the highest number being implementation-dependent, and possibly different between different group types. Debuggers should read back this field after writing to confirm they are using a hart group that is supported. If groups aren't implemented, then this entire field is 0.	WARL	preset

Field	Description	Access	Reset
[dmDmcsTwoHgwrite]# |hgwrite|	When 1 is written and is 0, for every selected hart the DM will change its group to the value written to , if the hardware supports that group for that hart. Implementations may also change the group of a minimal set of unselected harts in the same way, if that is necessary due to a hardware limitation. When 1 is written and is 1, the DM will change the group of the DM external trigger selected by to the value written to , if the hardware supports that group for that trigger. Writing 0 has no effect.	W1	-
|hgselect|	0 (harts): Operate on harts. 1 (triggers): Operate on DM external triggers. If there are no DM external triggers, this field must be tied to 0.	WARL	0

3.14.18. Halt Summary 0 (`haltsum0`, at 0x40)

[dmHaltsumZero]# Each bit in this read-only register indicates whether one specific hart is halted or not. Unavailable/nonexistent harts are not considered to be halted.

This register might not be present if fewer than 2 harts are connected to this DM.

The LSB reflects the halt status of hart \{hartsel[19:5],5'h0}, and the MSB reflects halt status of hart \{hartsel[19:5],5'h1f}.

This entire register is read-only.

31	0
latexmath:[$	haltsum0
$]	
32	

3.14.19. Halt Summary 1 (`haltsum1`, at 0x13)

[dmHaltsumOne]# Each bit in this read-only register indicates whether any of a group of harts is halted or not. Unavailable/nonexistent harts are not considered to be halted.

This register might not be present if fewer than 33 harts are connected to this DM.

The LSB reflects the halt status of harts \{hartsel[19:10],10'h0} through \{hartsel[19:10],10'h1f}. The MSB reflects the halt status of harts \{hartsel[19:10],10'h3e0} through \{hartsel[19:10],10'h3ff}.

This entire register is read-only.

31	0
latexmath:[$	haltsum1
$]	
32	

3.14.20. Halt Summary 2 (`haltsum2`, at 0x34)

[dmHaltsumTwo]# Each bit in this read-only register indicates whether any of a group of harts is halted or not. Unavailable/nonexistent harts are not considered to be halted.

This register might not be present if fewer than 1025 harts are connected to this DM.

The LSB reflects the halt status of harts \{hartsel[19:15],15'h0} through \{hartsel[19:15],15'h3ff}. The MSB reflects the halt status of harts \{hartsel[19:15],15'h7c00} through \{hartsel[19:15],15'h7fff}.

This entire register is read-only.

31	0
latexmath:[$	haltsum2
$]	
32	

3.14.21. Halt Summary 3 (`haltsum3`, at 0x35)

[dmHaltsumThree]# Each bit in this read-only register indicates whether any of a group of harts is halted or not. Unavailable/nonexistent harts are not considered to be halted.

This register might not be present if fewer than 32769 harts are connected to this DM.

The LSB reflects the halt status of harts 20'h0 through 20'h7fff. The MSB reflects the halt status of harts 20'hf8000 through 20'hfffff.

This entire register is read-only.

31	0
latexmath:[$	haltsum3
$]	
32	

3.14.22. System Bus Access Control and Status (`sbcs`, at 0x38)

[dmSbcs]#

31	29	28	23	22		21		20	
latexmath :[$	sbversion	$]		latexmath :[$	0	$]		latexmath :[$	sbbusyerr or
$]		latexmath :[$	sbbusy	$]		latexmath :[$	sbreadona ddr	$]	
3		6		1		1		1	

19	17	16		15		14	12	11	5
latexmath :[$	sbaccess	$]		latexmath :[$	sbautoinc rement	$]		latexmath :[$	sbreadond ata
$]		latexmath :[$	sberror	$]		latexmath :[$	sbasize	$]	
3		1		1		3		7	

4		3		2		1		0	
latexmath :[$	sbaccess12 8	$]		latexmath :[$	sbaccess6 4	$]		latexmath :[$	sbaccess3 2
$]		latexmath :[$	sbaccess1 6	$]		latexmath :[$	sbaccess8	$]	
1		1		1		1		1	

Field	Description	Access	Reset
Continued on next page			
[dmSbcsSbversion]# |sbversion|	0 (legacy): The System Bus interface conforms to mainline drafts of this spec older than 1 January, 2018. 1 (1.0): The System Bus interface conforms to this version of the spec. Other values are reserved for future versions.	R	1
[dmSbcsSbbusyerror]# |sbbusyerror|	Set when the debugger attempts to read data while a read is in progress, or when the debugger initiates a new access while one is already in progress (while is set). It remains set until it's explicitly cleared by the debugger. While this field is set, no more system bus accesses can be initiated by the Debug Module.	R/W1C	0

Field	Description	Access	Reset
[dmSbcsSbbusy]# |sbbusy|	When 1, indicates the system bus manager is busy. (Whether the system bus itself is busy is related, but not the same thing.) This bit goes high immediately when a read or write is requested for any reason, and does not go low until the access is fully completed. Writes to while is high result in undefined behavior. A debugger must not write to until it reads as 0.	R	0
[dmSbcsSbreadonaddr]# |sbreadonaddr|	When 1, every write to automatically triggers a system bus read at the new address.	R/W	0
[dmSbcsSbaccess]# |sbaccess|	Select the access size to use for system bus accesses. 0 (8bit): 8-bit 1 (16bit): 16-bit 2 (32bit): 32-bit 3 (64bit): 64-bit 4 (128bit): 128-bit If has an unsupported value when the DM starts a bus access, the access is not performed and is set to 4.	R/W	2
[dmSbcsSbautoincrement]# |sbautoincrement|	When 1, `sbaddress` is incremented by the access size (in bytes) selected in after every system bus access.	R/W	0
[dmSbcsSbreadondata]# |sbreadondata|	When 1, every read from automatically triggers a system bus read at the (possibly auto-incremented) address.	R/W	0

Field	Description	Access	Reset
[dmSbcsSberror]# |sberror|	When the Debug Module's system bus manager encounters an error, this field gets set. The bits in this field remain set until they are cleared by writing 1 to them. While this field is non-zero, no more system bus accesses can be initiated by the Debug Module. An implementation may report `Other" (7) for any error condition. 0 (none): There was no bus error. 1 (timeout): There was a timeout. 2 (address): A bad address was accessed. 3 (alignment): There was an alignment error. 4 (size): An access of unsupported size was requested. 7 (other): Other.	R/W1C	0
[dmSbcsSbasize]# |sbasize|	Width of system bus addresses in bits. (0 indicates there is no bus access support.)	R	Preset
[dmSbcsSbaccessOneTwentyeight]# |sbaccess128|	1 when 128-bit system bus accesses are supported.	R	Preset
[dmSbcsSbaccessSixtyfour]# |sbaccess64|	1 when 64-bit system bus accesses are supported.	R	Preset
[dmSbcsSbaccessThirtytwo]# |sbaccess32|	1 when 32-bit system bus accesses are supported.	R	Preset
[dmSbcsSbaccessSixteen]# |sbaccess16|	1 when 16-bit system bus accesses are supported.	R	Preset
|sbaccess8|	1 when 8-bit system bus accesses are supported.	R	Preset

3.14.23. System Bus Address 31:0 (`sbaddress0`, at 0x39)

[dmSbaddressZero]# If is 0, then this register is not present.

When the system bus manager is busy, writes to this register will set and don't do anything else.

If is 0, is 0, and is set then writes to this register start the following:

Set .

Perform a bus read from the new value of `sbaddress`.

If the read succeeded and is set, increment `sbaddress`.

Clear .

31	0
latexmath:[$	address
$]	
32	

Field	Description	Access	Reset
Continued on next page			
|address|	Accesses bits 31:0 of the physical address in `sbaddress`.	R/W	0

3.14.24. System Bus Address 63:32 (`sbaddress1`, at 0x3a)

[dmSbaddressOne]# If is less than 33, then this register is not present.

When the system bus manager is busy, writes to this register will set and don't do anything else.

31	0
latexmath:[$	address
$]	
32	

Field	Description	Access	Reset
Continued on next page			
|address|	Accesses bits 63:32 of the physical address in `sbaddress` (if the system address bus is that wide).	R/W	0

3.14.25. System Bus Address 95:64 (`sbaddress2`, at 0x3b)

[dmSbaddressTwo]# If is less than 65, then this register is not present.

When the system bus manager is busy, writes to this register will set and don't do anything else.

31	0
latexmath:[$	address
$]	
32	

Field	Description	Access	Reset
Continued on next page			
|address|	Accesses bits 95:64 of the physical address in `sbaddress` (if the system address bus is that wide).	R/W	0

3.14.26. System Bus Address 127:96 (`sbaddress3`, at 0x37)

[dmSbaddressThree]# If is less than 97, then this register is not present.

When the system bus manager is busy, writes to this register will set and don't do anything else.

31	0
latexmath:[$	address
$]	
32	

Field	Description	Access	Reset
Continued on next page			
|address|	Accesses bits 127:96 of the physical address in `sbaddress` (if the system address bus is that wide).	R/W	0

3.14.27. System Bus Data 31:0 (`sbdata0`, at 0x3c)

[dmSbdataZero]# If all of the `sbaccess` bits in are 0, then this register is not present.

Any successful system bus read updates `sbdata`. If the width of the read access is less than the width of `sbdata`, the contents of the remaining high bits may take on any value.

If either or isn't 0 then accesses do nothing.

If the bus manager is busy then accesses set , and don't do anything else.

Writes to this register start the following:

Set .

Perform a bus write of the new value of `sbdata` to `sbaddress`.

If the write succeeded and is set, increment `sbaddress`.

Clear .

Reads from this register start the following:

``Return'' the data.

Set .

If is set:

Perform a system bus read from the address contained in `sbaddress`, placing the result in `sbdata`.

If is set and the read was successful, increment `sbaddress`.

Clear .

Only has this behavior. The other `sbdata` registers have no side effects. On systems that have buses wider than 32 bits, a debugger should access after accessing the other `sbdata` registers.

31	0
latexmath:[$	data
$]	
32	

Field	Description	Access	Reset
Continued on next page			
|data|	Accesses bits 31:0 of `sbdata`.	R/W	0

3.14.28. System Bus Data 63:32 (`sbdata1`, at 0x3d)

[dmSbdataOne]# If and are 0, then this register is not present.

If the bus manager is busy then accesses set , and don't do anything else.

31	0
latexmath:[$	data
$]	
32	

Field	Description	Access	Reset
Continued on next page			
|data|	Accesses bits 63:32 of `sbdata` (if the system bus is that wide).	R/W	0

3.14.29. System Bus Data 95:64 (`sbdata2`, at 0x3e)

[dmSbdataTwo]# This register only exists if is 1.

If the bus manager is busy then accesses set , and don't do anything else.

31	0
latexmath:[$	data
$]	
32	

Field	Description	Access	Reset
Continued on next page			
|data|	Accesses bits 95:64 of `sbdata` (if the system bus is that wide).	R/W	0

3.14.30. System Bus Data 127:96 (`sbdata3`, at 0x3f)

[dmSbdataThree]# This register only exists if is 1.

If the bus manager is busy then accesses set , and don't do anything else.

31	0
latexmath:[$	data
$]	
32	

Field	Description	Access	Reset
Continued on next page			
|data|	Accesses bits 127:96 of `sbdata` (if the system bus is that wide).	R/W	0

3.14.31. Custom Features (`custom`, at 0x1f)

[dmCustom]# This optional register may be used for non-standard features. Future version of the debug spec will not use this address.

3.14.32. Custom Features 0 (`custom0`, at 0x70)

[dmCustomZero]# The optional through registers may be used for non-standard features. Future versions of the debug spec will not use these addresses.

Chapter 4. Sdext (ISA Extension)

This chapter describes the Sdext ISA extension. It must be implemented to make external debug work, and is only useful in conjunction with external debug.

Modifications to the RISC-V core to support debug are kept to a minimum. There is a special execution mode (Debug Mode) and a few extra CSRs. The DM takes care of the rest.

In order to be compatible with this specification an implementation must implement everything described in this chapter that is not explicitly listed as optional.

If Sdext is implemented and Sdtrig is not implemented, then accessing any of the Sdtrig CSRs must raise an illegal instruction exception.

4.1. Debug Mode

Debug Mode is a special processor mode used only when a hart is halted for external debugging. Because the hart is halted, there is no forward progress in the normal instruction stream. How Debug Mode is implemented is not specified here.

When executing code due to an abstract command, the hart stays in Debug Mode and the following apply:

All implemented instructions operate just as they do in M-mode, unless an exception is mentioned in this list.

All operations are executed with machine mode privilege, except that additional Debug Mode CSRs are accessible and in may be ignored according to . Full permission checks, or a relaxed set of permission checks, will apply according to .

All interrupts (including NMI) are masked.

Traps don't take place. Instead, they end execution of the program buffer and the hart remains in Debug Mode. Because they do not trap to M-mode, they do not update registers such as , `mepc`, `mcause`, ` mtval`, `mtval2`, and `mtinst`. The same is true for the equivalent privileged registers that are updated when trapping to other modes. Registers that may be updated as part of execution before the exception are allowed to be updated. For example, vector load/store instructions which raise exceptions may partially update the destination register and set `vstart` appropriately.

Triggers don't match or fire.

If is 0 then counters continue. If it is 1 then counters are stopped.

If is 0 then continues to update. If it is 1 then will not update. It will resynchronize with after leaving Debug Mode.

Instructions that place the hart into a stalled state act as a `nop`. This includes `wfi`, `wrs.sto`, and `wrs.nto`.

Almost all instructions that change the privilege mode have behavior. This includes `ecall`, `mret`, `sret`, and `uret`. (To change the privilege mode, the debugger can write and in). The only exception is `ebreak`, which ends execution of the Program Buffer when executed.

All control transfer instructions may act as illegal instructions if their destination is in the Program Buffer. If one such instruction acts as an illegal instruction, all such instructions must act as illegal instructions.

All control transfer instructions may act as illegal instructions if their destination is outside the Program Buffer. If one such instruction acts as an illegal instruction, all such instructions must act as illegal instructions.

Instructions that depend on the value of the PC (e.g. `auipc`) may act as illegal instructions.

Effective XLEN is DXLEN.

Forward progress is guaranteed.

When , the external debugger can set MPRV and MPP appropriately to have hardware perform memory accesses with the appropriate endianness, address translation, permission checks, and PMP/PMA checks (subject to). This is also the only way to access all of physical memory when 34-bit physical addresses are supported on a Sv32 hart. If hardware ties to 0 then the external debugger is expected to simulate all the effects of MPRV, including any extensions that affect memory accesses. For these reasons it is recommended to tie to 1.

4.2. Load-Reserved/Store-Conditional Instructions

The reservation registered by an `lr` instruction on a memory address may be lost when entering Debug Mode or while in Debug Mode. This means that there may be no forward progress if Debug Mode is entered between `lr` and `sc` pairs.

This is a behavior that debug users must be aware of. If they have a breakpoint set between a `lr` and `sc` pair, or are stepping through such code, the `sc` may never succeed. Fortunately in general use there will be very few instructions in such a sequence, and anybody debugging it will quickly notice that the reservation is not occurring. The solution in that case is to set a breakpoint on the first instruction after the `sc` and run to it. A higher level debugger may choose to automate this.

4.3. Wait for Interrupt Instruction

If halt is requested while `wfi` is executing, then the hart must leave the stalled state, completing this instruction's execution, and then enter Debug Mode.

4.4. Wait-on-Reservation-Set Instructions

If halt is requested while `wrs.sto` or `wrs.nto` is executing, then the hart must leave the stalled state, completing this instruction's execution, and then enter Debug Mode.

4.5. Single Step

4.5.1. Step Bit In Dcsr

This method is only available to external debuggers, and is the preferred way to single step.

An external debugger can cause a halted hart to execute a single instruction or trap and then re-enter

Debug Mode by setting before resuming. If is set when a hart resumes then it will single step, regardless of the reason for resuming.

If control is transferred to a trap handler while executing the instruction, then Debug Mode is re-entered immediately after the PC is changed to the trap handler, and the appropriate `tval` and `cause` registers are updated. In this case none of the trap handler is executed, and if the cause was a pending interrupt no instructions might be executed at all.

If executing or fetching the instruction causes a trigger to fire with action=1, Debug Mode is re-entered immediately after that trigger has fired. In that case is set to 2 (trigger) instead of 4 (single step). Whether the instruction is executed or not depends on the specific configuration of the trigger.

If the instruction that is executed causes the PC to change to an address where an instruction fetch causes an exception, that exception does not occur until the next time the hart is resumed. Similarly, a trigger at the new address does not fire until the hart actually attempts to execute that instruction.

If the instruction being stepped over would normally stall the hart, then instead the instruction is treated as a `nop`. This includes `wfi`, `wrs.sto`, and `wrs.nto`.

4.5.2. Icount Trigger

Native debuggers won't have access to , but can use the trigger by setting to 1.

This approach does have some limitations:

Interrupts will fire as usual. Debuggers that want to disable interrupts while stepping must disable them by changing , and specially handle instructions that read .

`wfi` instructions are not treated specially and might take a very long time to complete.

This mechanism cleanly supports a system which supports multiple privilege levels, where the OS or a debug stub runs in M-Mode while the program being debugged runs in a less privileged mode. Systems that only support M-Mode can use as well, but count must be able to count several instructions (depending on the software implementation). See Section [nativestep].

4.6. Reset

If the halt signal (driven by the hart's halt request bit in the Debug Module) or are asserted when a hart comes out of reset, the hart must enter Debug Mode before executing any instructions, but after performing any initialization that would usually happen before the first instruction is executed.

4.7. Halt

When a hart halts:

is updated.

and are set to reflect current privilege mode.

is set to the next instruction that should be executed.

If the current instruction can be partially executed and should be restarted to complete, then the

relevant state for that is updated. E.g. if a halt occurs during a partially executed vector instruction, then `vstart` is updated, and is updated to the address of the partially executed instruction. This is analogous to how vector instructions behave for exceptions.

The hart enters Debug Mode.

4.8. Resume

When a hart resumes:

changes to the value stored in .

The current privilege mode and virtualization mode are changed to that specified by and .

If the new privilege mode is less privileged than M-mode, in is cleared.

The hart is no longer in debug mode.

4.9. XLEN

While in Debug Mode, XLEN is DXLEN. It is up to the debugger to determine the XLEN during normal program execution (by looking at) and to clearly communicate this to the user.

4.10. Core Debug Registers

The supported Core Debug Registers must be implemented for each hart that can be debugged. They are CSRs, accessible using the RISC-V `csr` opcodes and optionally also using abstract debug commands.

Attempts to access an unimplemented Core Debug Register raise an illegal instruction exception.

4.11. Virtual Debug Registers

Chapter 5. Sdtrig (ISA Extension)

This chapter describes the Sdtrig ISA extension, which can be implemented independently of functionality described in the other chapters. It consists exclusively of the Trigger Module ™.

Triggers can cause a breakpoint exception, entry into Debug Mode, or a trace action without having to execute a special instruction. This makes them invaluable when debugging code from ROM. They can trigger on execution of instructions at a given memory address, or on the address/data in loads/stores.

If Sdtrig is implemented, the Trigger Module must support at least one trigger. Accessing trigger CSRs that are not used by any of the implemented triggers must result in an illegal instruction exception. M-Mode and Debug Mode accesses to trigger CSRs that are used by any of the implemented triggers must succeed, regardless of the current type of the currently selected trigger.

A trigger matches when the conditions that it specifies (e.g. a load from a specific address) are met. A trigger fires when a trigger that matches performs the action configured for that trigger.

Triggers do not fire while in Debug Mode.

5.1. Enumeration

Each trigger may support a variety of features. A debugger can build a list of all triggers and their features as follows:

Write 0 to . If this results in an illegal instruction exception, then there are no triggers implemented.

Read back and check that it contains the written value. If not, exit the loop.

Read .

If that caused an exception, the debugger must read to discover the type. (If is 0, this trigger doesn't exist. Exit the loop.)

If is 1, this trigger doesn't exist. Exit the loop.

Otherwise, the selected trigger supports the types discovered in .

Repeat, incrementing the value in .

The above algorithm reads back so that implementations which have 2^n triggers only need to implement n bits of .

The algorithm checks and in case the implementation has m bits of but fewer than 2^m triggers.

5.2. Actions

Triggers can be configured to take one of several actions when they fire. Table #tab:action lists all options.

|r|L| Value & Description
& Raise a breakpoint exception. (Used when software wants to use the trigger module without an external debugger attached.) must contain the virtual address of the next instruction that must be

executed to preserve the program flow.
& Enter Debug Mode. must contain the virtual address of the next instruction that must be executed to preserve the program flow.

This action is only legal when the trigger's is 1. Since the `tdata` registers are WARL, hardware should clear the action field whenever the action field is 1, the new value of would be 0, and the new value of the action field would be 1.

This action can only be supported if Sdext is implemented on the hart.
& Trace on, described in the trace specification.
& Trace off, described in the trace specification.
& Trace notify, described in the trace specification.
& Reserved for use by the trace specification.
– 9 & Send a signal to TM external trigger output 0 or 1 (respectively).
other & Reserved for future use.

Actions 8 and 9 are intended to increment custom event counters, but these signals could also be brought to outputs for use by external logic.

5.3. Priority

Table #tab:priority lists the synchronous exceptions from the Privileged Spec, and where the various types of triggers fit in. The first 3 columns come from the Privileged Spec, and the final column shows where triggers fit in. Priorities in the table are separated by horizontal lines, so e.g. etrigger and itrigger have the same priority. If this table contradicts the table in the Privileged Spec, then the latter takes precedence.

This table only applies if triggers are precise. Otherwise triggers will fire some indeterminate time after the event, and the priority is irrelevant. When triggers are chained, the priority is the lowest priority of the triggers in the chain.

|l|p.7in|p2.3in|p2.5in| Priority & Exception Code & Description & Trigger
Highest & 3 & & etrigger
& 3 & & icount
& 3 & & itrigger
& 3 & & mcontrol/mcontrol6 after (on previous instruction)
& 3 & Instruction address breakpoint & mcontrol/mcontrol6 execute address before
& 12, 20, 1 & During instruction address translation: First encountered page fault, guest-page fault, or access fault &
& 1 & With physical address for instruction: Instruction access fault &
& 3 & & mcontrol/mcontrol6 execute data before
& 2 & Illegal instruction &
& 22 & Virtual instruction &
& 0 & Instruction address misaligned &
& 8, 9, 10, 11 & Environment call &
& 3 & Environment break &
& 3 & Load/Store/AMO address breakpoint & mcontrol/mcontrol6 load/store address/data before
& 4, 6 & Optionally: Load/Store/AMO address misaligned &
& 13, 15, 21, 23, 5, 7 & During address translation for an explicit memory access: First encountered page fault, guest-page fault, or access fault &
& 5, 7 & With physical address for an explicit memory access: Load/store/AMO access fault &
& 4, 6 & If not higher priority: Load/store/AMO address misaligned &

Lowest & 3 & & mcontrol/mcontrol6 load data before

When multiple triggers in the same priority fire at once, (if implemented) is set for all of them. If more than one of these triggers has then `tval` is updated in accordance with one of them, but which one is . If one of these triggers has the `enter Debug Mode'' action (1) and another trigger has the` raise a breakpoint exception" action (0), the preferred behavior is to have both actions take place. It is implementation-dependent which of the two happens first. This ensures both that the presence of an external debugger doesn't affect execution and that a trigger set by user code doesn't affect the external debugger. If this is not implemented, then the hart must enter Debug Mode and ignore the breakpoint exception. In the latter case, of the trigger whose action is 0 must still be set, giving a debugger an opportunity to handle this case. What happens with trace actions when triggers with different actions are also firing is left to the trace specification.

5.4. Native Triggers

Triggers can be used for native debugging when . If supported by the hart and desired by the debugger, triggers will often be programmed to have so that when they fire they cause a breakpoint exception to trap to a more privileged mode. That breakpoint exception can either be taken in M-mode or it can be delegated to a less privileged mode. However, it is possible for triggers to fire in the same mode that the resulting exception will be handled in.

In these cases such a trigger may cause a breakpoint exception while already in a trap handler. This might leave the hart unable to resume normal execution because state such as and would be overwritten.

In particular, when :

mcontrol and mcontrol6 triggers with can cause a breakpoint exception that is taken from M-mode to M-mode (regardless of delegation).

mcontrol and mcontrol6 triggers with can cause a breakpoint exception that is taken from S-mode to S-mode if .

mcontrol6 triggers with can cause a breakpoint exception that is taken from VS-mode to VS-mode if and .

icount triggers with can cause a breakpoint exception that is taken from M-mode to M-mode (regardless of delegation).

icount triggers with can cause a breakpoint exception that is taken from S-mode to S-mode if .

icount triggers with can cause a breakpoint exception that is taken from VS-mode to VS-mode if and .

etrigger and itrigger triggers will always be taken from a trap handler before the first instruction of the handler. If etrigger/itrigger is set to trigger on exception/interrupt X and if X is delegated to mode Y then the trigger will cause a breakpoint exception that is taken from mode Y to mode Y unless breakpoint exceptions are delegated to a more privileged mode than Y.

tmexttrigger triggers are asynchronous and may occur in any mode and at any time.

Harts that support triggers with should implement one of the following two solutions to solve the problem of reentrancy:

The hardware prevents triggers with from matching or firing while in M-mode and while in is 0. If then it prevents triggers with from matching or firing while in S-mode and while in is 0. If and then it prevents triggers with from matching or firing while in VS-mode and while in is 0.

and in is implemented. is hard-wired to 0.

The first option has the limitation that interrupts might be disabled at times when a user still might want triggers to fire. It has the benefit that breakpoints are not required to be handled in M-mode.

The second option has the benefit that it only disables triggers during the trap handler, though it requires specific software support for this debug feature in the M-mode trap handlers. It can only work if breakpoints are not delegated to less privileged modes and therefore targets primarily implementations without S-mode.

Because is not accessible to S-mode, the second option can not be extended to accommodate delegation without adding additional S-mode and VS-mode CSRs.

Both options prevent etrigger and itrigger from having any effect on exceptions and interrupts that are handled in M-mode. They also prevent triggering during some initial portion of each handler. Debuggers should use other mechanisms to debug these cases, such as patching the handler or setting a breakpoint on the instruction after is cleared.

5.5. Memory Access Triggers

and both enable triggers on memory accesses. This section describes for both of them how certain corner cases are treated.

5.5.1. A Extension

If the A extension is supported, then triggers on loads/stores treat them as follows:

`lr` instructions are loads.

Successful `sc` instructions are stores.

It is whether failing `sc` instructions are stores or not.

Each AMO instruction is a load for the read portion of the operation. The address is always available to trigger on, although the value loaded might not be, depending on the hardware implementation.

Each AMO instruction is a store for the write portion of the operation. The address is always available to trigger on, although the value stored might not be, depending on the hardware implementation.

If the destination register of any load or AMO is then it is whether a data load trigger will match. Whether data store triggers match on AMOs is .

5.5.2. Combined Accesses

Some instructions lead a hart to perform multiple memory accesses. This includes vector loads and stores, as well as `cm.push` and `cm.pop` instructions. The Trigger Module should match such accesses as if they all happened individually. E.g. a vector load should be treated as if it performed multiple loads of size SEW (selected element width), and `cm.push` should be treated as if it performed multiple stores of size XLEN.

5.5.3. Cache Operations

Cache operations are infrequently performed, and code that uses them can have hard-to-find bugs. For the purposes of debug triggers, two classes of cache operations must match as stores:

Cache operations that enable software to maintain coherence between otherwise non-coherent implicit and explicit memory accesses.

Cache operations that perform block writes of constant data.

Only triggers with and will match. Since cache operations affect multiple addresses, there are multiple possible values to compare against. Implementations must implement one of the following options. From most desirable to least desirable, they are:

Every address from the effective address rounded down to the nearest cache block boundary (inclusive) to the effective address rounded up to the nearest cache block boundary (exclusive) is a compare value.

The effective address rounded down to the nearest cache block boundary is a compare value.

The effective address of the instruction is a compare value.

Cache operations encoded as HINTs do not match debug triggers.

The above language intends to capture the trigger behavior with respect to the cache operations to be introduced in a forthcoming I/D consistency extension.

For RISC-V Base Cache Management Operation ISA Extensions 1.0.1, this means the following:

, , and match as if they are stores because they affect consistency.

matches as if it is a store because it performs a block write of constant data.

The prefetch instructions don't match at all.

5.6. Multiple State Change Instructions

An instruction that performs multiple architectural state changes (e.g., register updates and/or memory accesses) might cause a trigger to fire at an intermediate point in its execution. As a result, architectural state changes up to that point might have been performed, while subsequent state changes, starting from the event that activated the trigger, might not have been. The definition of such an instruction will specify the order in which architectural state changes take place. Alternatively, it may state that partial execution is not allowed, implying that a mid-execution trigger must prevent any architectural state changes from occurring.

Debuggers won't be aware if an instruction has been partially executed. When they resume execution, they will execute the same instruction once more. Therefore, it's crucial that partially executing the instruction and then executing it again leaves the hart in a state closely resembling the state it would have been in if the instruction had only been executed once.

5.7. Trigger Registers

These registers are CSRs, accessible using the RISC-V `csr` opcodes and optionally also using abstract debug commands.

Almost all trigger functionality is optional. All `tdata` registers follow write-any-read-legal semantics. If a debugger writes an unsupported configuration, the register will read back a value that is supported (which may simply be a disabled trigger). This means that a debugger must always read back values it writes to `tdata` registers, unless it already knows already what is supported. Writes to one `tdata` register must not modify the contents of other `tdata` registers, nor the configuration of any trigger besides the one that is currently selected.

The combination of these rules means that a debugger cannot simply set a trigger by writing , then , etc. The current value of might not be legal with the new value of . To help with this situation, it is guaranteed that writing 0 to disables the trigger, and leaves it in a state where and can be written with any value that makes sense for any trigger type supported by this trigger.

As a result, a debugger can write any supported trigger as follows:

Write 0 to . (This will result in containing a non-zero value, since the register is .)

Write desired values to and .

Write desired value to .

Code that restores CSR context of triggers that might be configured to fire in the current privilege mode must use this same sequence to restore the triggers. This avoids the problem of a partially written trigger firing at a different time than is expected.

Attempts to access an unimplemented Trigger Register raise an illegal instruction exception.

Chapter 6. JTAG Debug Transport Module

This Debug Transport Module is based around a normal JTAG Test Access Port (TAP). The JTAG TAP allows access to arbitrary JTAG registers by first selecting one using the JTAG instruction register (IR), and then accessing it through the JTAG data register (DR).

6.1. JTAG Background

JTAG refers to IEEE Std 1149.1-2013. It is a standard that defines test logic that can be included in an integrated circuit to test the interconnections between integrated circuits, test the integrated circuit itself, and observe or modify circuit activity during the component's normal operation. This specification uses the latter functionality. The JTAG standard defines a Test Access Port (TAP) that can be used to read and write a few custom registers, which can be used to communicate with debug hardware in a component.

6.2. JTAG DTM Registers

JTAG TAPs used as a DTM must have an IR of at least 5 bits. When the TAP is reset, IR must default to 00001, selecting the IDCODE instruction. A full list of JTAG registers along with their encoding is in Table #dtmTable:jtagregisters. If the IR actually has more than 5 bits, then the encodings in Table #dtmTable:jtagregisters should be extended with 0's in their most significant bits, except for the 0x1f encoding of BYPASS, which must be extended with 1's in the most significant bits. The only regular JTAG registers a debugger might use are BYPASS and IDCODE, but this specification leaves IR space for many other standard JTAG instructions. Unimplemented instructions must select the BYPASS register.

6.3. JTAG Connector

6.3.1. Recommended JTAG Connector

To make it easy to acquire debug hardware, this spec recommends a connector that is compatible with the MIPI-10 .05 inch connector specification, as described in MIPI Debug & Trace Connector Recommendations, Version 1.20, 2 July 2021.

The connector has .05 inch spacing, gold-plated male header with .016 inch thick hardened copper or beryllium bronze square posts (SAMTEC FTSH or equivalent). Female connectors are compatible $20\mu m$ gold connectors.

Viewing the male header from above (the pins pointing at your eye), a target's connector looks as it does in Table #tab:mipiten[1]. The function of each pin is described in Table #tab:pinout.

VREF DEBUG	1	2	TMS
GND	3	4	TCK
GND	5	6	TDO
GND or KEY	7	8	TDI
GND	9	10	nRESET

Table 3. MIPI 10-pin JTAG + nRESET Connector Diagram

|l|c|L| Essential & GND & Connected to ground.
& TCK & JTAG TCK signal, driven by the debug adapter.
& TDI & JTAG TDI signal, driven by the debug adapter.
& TDO & JTAG TDO signal, driven by the target.
& TMS & JTAG TMS signal, driven by the debug adapter.
& VREF DEBUG & Reference voltage for logic high.
Recommended & nRESET & Open drain active low reset signal, usually driven by the debug adapter. The signal may be used bi-directional to drive or sense the target reset signal.

Asserting reset should reset any RISC-V cores as well as any other peripherals on the PCB. It should not reset the debug logic. This pin is optional but strongly encouraged.

nRESET should never be connected to the TAP reset, otherwise the debugger might not be able to debug through a reset to discover the cause of a crash or to maintain execution control after the reset.
& KEY & This pin may be cut on the male and plugged on the female header to ensure the header is always plugged in correctly. It is, however, recommended to use this pin as an additional ground, to allow for fastest TCK speeds. A shrouded connector should be used to prevent the cable from being plugged in incorrectly.
Advanced & EXT & Reserved for custom use. Could be an input or an output.
& TRIGIN & Not used by this specification, to be driven by debug adapter. (Can be used for extended functions like UART or boot mode selection by some debug adapters).
& TRIGOUT & Not used by this specification, driven by the target.
Specialized & nTRST & Test reset, driven by the debug adapter. Asserting nTRST initializes the JTAG DTM asynchronously. It is used in systems where the JTAG DTM is not ready to be used after a normal power up. This signal is sometimes called TRST*.
Legacy & RTCK & Return test clock, driven by the target. A target may relay the TCK signal here once it has processed it, allowing a debugger to adjust its TCK frequency in response.

This signal should only be used to support legacy components that rely on this functionality.
& nTRST_PD & Test reset pull-down, driven by the debug adapter. Same function as nTRST, but with pull-down resistor on target.

This signal should only be used to support legacy components that rely on this functionality.

If a hardware platform requires nTRST then it is permissible to reuse the nRESET pin as the nTRST signal, resulting in a MIPI 10-pin JTAG
nTRST connector.

6.3.2. Alternate JTAG Connector

The MIPI-10 connector should provide plenty of signals for all modern hardware. If a design does need legacy JTAG signals, then the MIPI-20 connector should be used. Pins whose functionality isn't needed may be left unconnected.

Its physical connector is virtually identical to MIPI-10, except that it's twice as long, supporting twice as many pins. Its pinout is shown in Table #tab:mipitwenty[2]. The function of each pin is described in Table #tab:pinout.

VREF DEBUG	1	2	TMS
GND	3	4	TCK
GND	5	6	TDO

VREF DEBUG	1	2	TMS
GND or KEY	7	8	TDI
GND	9	10	nRESET
GND	11	12	GND or RTCK
GND	13	14	NC or nTRST_PD
GND	15	16	nTRST or NC
GND	17	18	TRIGIN or NC
GND	19	20	TRIGOUT or GND

Table 4. MIPI 20-pin JTAG Connector Diagram

6.4. cJTAG

This spec does not have specific recommendations on how to use the cJTAG protocol.

When implementing cJTAG access to a JTAG DTM, the MIPI 10-pin Narrow JTAG connector should be used. Pins whose functionality isn't needed may be left unconnected.

Viewing the male header from above (the pins pointing at your eye), a target's connector looks as it does in Table #tab:mipicjtag[3].

VREF DEBUG	1	2	TMSC
GND	3	4	TCKC
GND	5	6	EXT or NC
GND or KEY	7	8	NC or nTRST_PD
GND	9	10	nRESET

Table 5. MIPI 10-pin Narrow JTAG Connector Diagram

Chapter 7. Hardware Implementations

Below are two possible implementations. A designer could choose one, mix and match, or come up with their own design.

7.1. Abstract Command Based

Halting happens by stalling the hart execution pipeline.

Muxes on the register file(s) allow for accessing GPRs and CSRs using the Access Register abstract command.

Memory is accessed using the Abstract Access Memory command or through System Bus Access.

This implementation could allow a debugger to collect information from the hart even when that hart is unable to execute instructions.

7.2. Execution Based

This implementation only implements the Access Register abstract command for GPRs on a halted hart, and relies on the Program Buffer for all other operations. It uses the hart's existing pipeline and ability to execute from arbitrary memory locations to avoid modifications to a hart's datapath.

When the halt request bit is set, the Debug Module raises a special interrupt to the selected harts. This interrupt causes each hart to enter Debug Mode and jump to a defined memory region that is serviced by the DM and is only accessible to the harts in Debug Mode. Accesses to this memory should be uncached to avoid side effects from debugging operations. When taking this jump, is saved to and is updated in . This jump is similar to a trap but it is not architecturally considered a trap, so for instance doesn't count as a trap for trigger behavior.

The code in the Debug Module causes the hart to execute a `park loop." In the park loop the hart writes its to a memory location within the Debug Module to indicate that it is halted. To allow the DM to individually control one out of several halted harts, each hart polls for flags in a DM-controlled memory location to determine whether the debugger wants it to execute the Program Buffer or perform a resume.

To execute an abstract command, the DM first populates some internal words of program buffer according to . When is set, the DM populates these words with `lw <gpr>, 0x400(zero)` or `sw 0x400(zero), <gpr>`. 64- and 128-bit accesses use `ld`/`sd` and `lq`/`sq` respectively. If is not set, the DM populates these instructions as `nop`s. If is set, execution continues to the debugger-controlled Program Buffer, otherwise the DM causes a `ebreak` to execute immediately.

When `ebreak` is executed (indicating the end of the Program Buffer code) the hart returns to its park loop. If an exception is encountered, the hart jumps to an address within the Debug Module. The code there causes the hart to write to the Debug Module indicating an exception. Then the hart jumps back to the park loop. The DM infers from the write that there was an exception, and sets appropriately. Typically the hart will execute a `fence` instruction before entering the park loop, to ensure that any effects from the abstract command, such as a write to , take effect before the DM returns to 0.

To resume execution, the debug module sets a flag which causes the hart to execute a `dret`. `dret` is an instruction that only has meaning while in Debug Mode and not executing from the Program Buffer.

Its recommended encoding is 0x7b200073. When `dret` is executed, is restored from and normal execution resumes at the privilege set by and .

etc. are mapped into regular memory at an address relative to with only a 12-bit `imm`. The exact address is an implementation detail that a debugger must not rely on. For example, the `data` registers might be mapped to `0x400`.

For additional flexibility, , etc. are mapped into regular memory immediately preceding , in order to form a contiguous region of memory which can be used for either program execution or data transfer.

The PMP must not disallow fetches, loads, or stores in the address range associated with the Debug Module when the hart is in Debug Mode, regardless of how the PMP is configured. The same is true of PMA. Without this guarantee, the park loop would enter an infinite loop of traps and debug would not be possible.

7.3. Debug Module Interface Signals

As stated in section [dmi] the details of the DMI are left to the system designer. It is quite often the case that only one DTM and one DM is implemented. In this case it might be useful to comply with the signals suggested in table #tab:dmi_signals[1.1], which is the implementation used in the open-source rocket-chip RISC-V core.

The DTM can start a request when the DM sets REQ_READY to 1. When this is the case REQ_OP can be set to 1 for a read or 2 for a write request. The desired address is driven with the REQ_ADDRESS signal. Finally REQ_VALID is set high, indicating to the DM that a valid request is pending.

The DM must respond to a request from the DTM when RSP_READY is high. The status of the response is indicated by the RSP_OP signal (see). The data of the response is driven to RSP_DATA. A pending response is signalled by setting RSP_VALID.

Signal	Width	Source	Description
REQ_VALID	1	DTM	Indicates that a valid request is pending
REQ_READY	1	DM	Indicates that the DM is able to process a request
REQ_ADDRESS		DTM	Requested address
REQ_DATA	32	DTM	Requested data
REQ_OP	2	DTM	Same meaning as the field
RSP_VALID	1	DM	Indicates that a valid respond is pending
RSP_READY	1	DTM	Indicates that the DTM is able to process a respond
RSP_DATA	32	DM	Response data
RSP_OP	2	DM	Same meaning as the field

Table 6. Signals for the suggested DMI between one DTM and one DM

Chapter 8. Debugger Implementation

8.1. C Header File

github.com/riscv/riscv-debug-spec contains instructions for generating a C header file that defines macros for every field in every register/abstract command mentioned in this document.

8.2. External Debugger Implementation

This section details how an external debugger might use the described debug interface to perform some common operations on RISC-V cores using the JTAG DTM described in Section #sec:jtagdtm. All these examples assume a 32-bit core but it should be easy to adapt the examples to 64- or 128-bit cores.

To keep the examples readable, they all assume that everything succeeds, and that they complete faster than the debugger can perform the next access. This will be the case in a typical JTAG setup. However, the debugger must always check the sticky error status bits after performing a sequence of actions. If it sees any that are set, then it should attempt the same actions again, possibly while adding in some delay, or explicit checks for status bits.

8.2.1. Debug Module Interface Access

To read an arbitrary Debug Module register, select , and scan in a value with set to 1, and set to the desired register address. In Update-DR the operation will start, and in Capture-DR its results will be captured into . If the operation didn't complete in time, will be 3 and the value in must be ignored. The busy condition must be cleared by writing in , and then the second scan scan must be performed again. This process must be repeated until returns 0. In later operations the debugger should allow for more time between Update-DR and Capture-DR.

To write an arbitrary Debug Bus register, select , and scan in a value with set to 2, and and set to the desired register address and data respectively. From then on everything happens exactly as with a read, except that a write is performed instead of the read.

It should almost never be necessary to scan IR, avoiding a big part of the inefficiency in typical JTAG use.

8.2.2. Checking for Halted Harts

A user will want to know as quickly as possible when a hart is halted (e.g. due to a breakpoint). To efficiently determine which harts are halted when there are many harts, the debugger uses the `haltsum` registers. Assuming the maximum number of harts exist, first it checks . For each bit set there, it writes , and checks . This process repeats through and . Depending on how many harts exist, the process should start at one of the lower `haltsum` registers.

8.2.3. Halting

To halt one or more harts, the debugger selects them, sets , and then waits for to indicate the harts are halted. Then it can clear to 0, or leave it high to catch a hart that resets while halted.

8.2.4. Running

First, the debugger should restore any registers that it has overwritten. Then it can let the selected harts run by setting . Once is set, the debugger knows the selected harts have resumed. Harts might halt very quickly after resuming (e.g. by hitting a software breakpoint) so the debugger cannot use /to check whether the hart resumed.

8.2.5. Single Step

Using the hardware single step feature is almost the same as regular running. The debugger just sets in before letting the hart run. The hart behaves exactly as in the running case, except that interrupts may be disabled (depending on) and it only fetches and executes a single instruction before re-entering Debug Mode.

8.2.6. Accessing Registers

8.2.6.1. Using Abstract Command

Read using abstract command:

Op	Address	Value	Comment
Write		= 2, , = 0x1008	Read
Read		-	Returns value that was in

Write using abstract command:

Op	Address	Value	Comment
Write		new value	
Write		= 2, , , = 0x300	Write

8.2.6.2. Using Program Buffer

Abstract commands are used to exchange data with GPRs. Using this mechanism, other registers can be accessed by moving their value into/out of GPRs.

Write using program buffer:

Op	Address	Value	Comment
Write		`csrw s0, MSTATUS`	
Write	`progbuf1`	`ebreak`	
Write		new value	
Write		= 2, , , , = 0x1008	Write , then execute program buffer

Read using program buffer:

Op	Address	Value	Comment
Write		`fmv.x.s s0, f1`	
Write	`progbuf1`	`ebreak`	

Op	Address	Value	Comment
Write			Execute program buffer
Write		, = 0x1008	read
Read		-	Returns the value that was in

8.2.7. Reading Memory

8.2.7.1. Using System Bus Access

With system bus access, addresses are physical system bus addresses.

Read a word from memory using system bus access:

Op	Address	Value	Comment
Write		= 2,	Setup
Write		address	
Read		-	Value read from memory

Read block of memory using system bus access:

Op	Address	Value	Comment
Write		= 2, , ,	Turn on autoread and autoincrement
Write		address	Writing address triggers read and increment
Read		-	Value read from memory
Read		-	Next value read from memory
...	...	...	...
Write		0	Disable autoread
Read		-	Get last value read from memory.

8.2.7.2. Using Program Buffer

Through the Program Buffer, the hart performs the memory accesses. Addresses are physical or virtual (depending on and other system configuration).

Read a word from memory using program buffer:

Op	Address	Value	Comment
Write		`lw s0, 0(s0)`	
Write	`progbuf1`	`ebreak`	
Write		address	
Write		, , , = 0x1008	Write , then execute program buffer

Op	Address	Value	Comment
Write		= 0x1008	Read
Read		-	Value read from memory

Read block of memory using program buffer:

Op	Address	Value	Comment
Write		`lw s1, 0(s0)`	
Write	`progbuf1`	`addi s0, s0, 4`	
Write	`progbuf2`	`ebreak`	
Write		address	
Write		, , , = 0x1008	Write , then execute program buffer
Write		, = 0x1009	Read , then execute program buffer
Write			Set
Read		-	Get value read from memory, then execute program buffer
Read		-	Get next value read from memory, then execute program buffer
...	...	...	...
Write		0	Clear
Read		-	Get last value read from memory.

8.2.7.3. Using Abstract Memory Access

Abstract memory accesses act as if they are performed by the hart, although the actual implementation may differ.

Read a word from memory using abstract memory access:

Op	Address	Value	Comment
Write		address	
Write		cmdtype=2,	
Read		-	Value read from memory

Read block of memory using abstract memory access:

Op	Address	Value	Comment
Write		1	Re-execute the command when is accessed
Write		address	
Write		cmdtype=2, ,	

Op	Address	Value	Comment
Read		-	Read value, and trigger reading of next address
...	...	...	...
Write		0	Disable auto-exec
Read		-	Get last value read from memory.

8.2.8. Writing Memory

8.2.8.1. Using System Bus Access

With system bus access, addresses are physical system bus addresses.

Write a word to memory using system bus access:

Op	Address	Value	Comment
Write		= 2	Configure access size
Write		address	
Write		value	

Write a block of memory using system bus access:

Op	Address	Value	Comment
Write		= 2,	Turn on autoincrement
Write		address	
Write		value0	
Write		value1	
...	...	...	...
Write		valueN	

8.2.8.2. Using Program Buffer

Through the Program Buffer, the hart performs the memory accesses. Addresses are physical or virtual (depending on and other system configuration).

Write a word to memory using program buffer:

Op	Address	Value	Comment
Write		`sw s1, 0(s0)`	
Write	`progbuf1`	`ebreak`	
Write		address	
Write		, , = 0x1008	Write
Write		value	
Write		, , , = 0x1009	Write , then execute program buffer

Write block of memory using program buffer:

Op	Address	Value	Comment
Write		`sw s1, 0(s0)`	
Write	`progbuf1`	`addi s0, s0, 4`	
Write	`progbuf2`	`ebreak`	
Write		address	
Write		, , = 0x1008	Write
Write		value0	
Write		, , , = 0x1009	Write , then execute program buffer
Write			Set
Write		value1	
...	...	...	...
Write		valueN	
Write		0	Clear

8.2.8.3. Using Abstract Memory Access

Abstract memory accesses act as if they are performed by the hart, although the actual implementation may differ.

Write a word to memory using abstract memory access:

Op	Address	Value	Comment
Write		address	
Write		value	
Write		cmdtype=2, , write=1	

Write a block of memory using abstract memory access:

Op	Address	Value	Comment
Write		address	
Write		value0	
Write		cmdtype=2, , write=1,	
Write		1	Re-execute the command when is accessed
Write		value1	
Write		value2	
...	...	...	...
Write		valueN	
Write		0	Disable auto-exec

8.2.9. Triggers

A debugger can use hardware triggers to halt a hart when a certain event occurs. Below are some examples, but as there is no requirement on the number of features of the triggers implemented by a hart, these examples might not be applicable to all implementations. When a debugger wants to set a trigger, it writes the desired configuration, and then reads back to see if that configuration is supported. All examples assume XLEN=32.

Enter Debug Mode when the instruction at 0x80001234 is executed, to be used as an instruction breakpoint in ROM:

|r|r|L| & 0x6980105c & type=6, dmode=1, action=1, select=0, match=0, m=1, s=1, u=1, vs=1, vu=1, execute=1
& 0x80001234 & address

Enter Debug Mode when performing a load at address 0x80007f80 in M-mode or S-mode or U-mode:

|r|r|L| & 0x68001059 & type=6, dmode=1, action=1, select=0, match=0, m=1, s=1, u=1, load=1
& 0x80007f80 & address

Enter Debug Mode when storing to an address between 0x80007c80 and 0x80007cef (inclusive) in VS-mode or VU-mode when hgatp.VMID=1:

|r|r|L| & 0x69801902 & type=6, dmode=1, action=1, chain=1, select=0, match=2, vs=1, vu=1, store=1
& 0x80007c80 & start address (inclusive)
& 0x03000000 & mhselect=6, mhvalue=0
& 0x69801182 & type=6, dmode=1, action=1, select=0, match=3, vs=1, vu=1, store=1
& 0x80007cf0 & end address (exclusive)
& 0x03000000 & mhselect=6, mhvalue=0

Enter Debug Mode when storing to an address between 0x81230000 and 0x8123ffff (inclusive):

|r|r|L| & 0x698010da & type=6, dmode=1, action=1, select=0, match=1, m=1, s=1, u=1, vs=1, vu=1, store=1
& 0x81237fff & 16 upper bits to match exactly, then 0, then all ones.

Enter Debug Mode when loading from an address between 0x86753090 and 0x8675309f or between 0x96753090 and 0x9675309f (inclusive):

|r|r|L| & 0x69801a59 & type=6, dmode=1, action=1, chain=1, match=4, m=1, s=1, u=1, vs=1, vu=1, load=1
& 0xfff03090 & Mask for low half, then match for low half
& 0x698012d9 & type=6, dmode=1, action=1, match=5, m=1, s=1, u=1, vs=1, vu=1, load=1
& 0xefff8675 & Mask for high half, then match for high half

8.2.10. Handling Exceptions

Generally the debugger can avoid exceptions by being careful with the programs it writes. Sometimes they are unavoidable though, e.g. if the user asks to access memory or a CSR that is not implemented. A typical debugger will not know enough about the hardware platform to know what's going to happen, and must attempt the access to determine the outcome.

When an exception occurs while executing the Program Buffer, becomes set. The debugger can check this field to see whether a program encountered an exception. If there was an exception, it's left to the

debugger to know what must have caused it.

8.2.11. Quick Access

There are a variety of instructions to transfer data between GPRs and the ` data` registers. They are either loads/stores or CSR reads/writes. The specific addresses also vary. This is all specified in . The examples here use the pseudo-op `transfer dest, src` to represent all these options.

Halt the hart for a minimum amount of time to perform a single memory write:

Op	Address	Value	Comment
Write		`transfer arg2, s0`	Save
Write	`progbuf1`	`transfer s0, arg0`	Read first argument (address)
Write	`progbuf2`	`transfer arg0, s1`	Save
Write	`progbuf3`	`transfer s1, arg1`	Read second argument (data)
Write	`progbuf4`	`sw s1, 0(s0)`	
Write	`progbuf5`	`transfer s1, arg0`	Restore
Write	`progbuf6`	`transfer s0, arg2`	Restore
Write	`progbuf7`	`ebreak`	
Write		address	
Write	`data1`	data	
Write		0x10000000	Perform quick access

This shows an example of setting the bit in to enable a hardware breakpoint in M-mode. Similar quick access instructions could have been used previously to configure the trigger that is being enabled here:

Op	Address	Value	Comment
Write		`transfer arg0, s0`	Save
Write	`progbuf1`	`li s0, (1 << 6)`	Form the mask for bit
Write	`progbuf2`	`csrrs x0, , s0`	Apply the mask to
Write	`progbuf3`	`transfer s0, arg2`	Restore
Write	`progbuf4`	`ebreak`	
Write		0x10000000	Perform quick access

8.3. Native Debugger Implementation

The spec contains a few features to aid in writing a native debugger. This section describes how some common tasks might be achieved.

8.3.1. Single Step

Single step is straightforward if the OS or a debug stub runs in M-Mode while the program being debugged runs in a less privileged mode. When a step is required, the OS or debug stub writes , , before

returning control to the lower user program with an `mret` instruction.

Stepping code running in the same privilege mode as the debugger is more complicated, depending on what other debug features are implemented.

If hardware implements and , then stepping through non-trap code which doesn't allow for nested interrupts is also straightforward.

If hardware automatically prevents triggers from matching when entering a trap handler as described in Section #sec:nativetrigger, then a carefully written trap handler can ensure that interrupts are disabled whenever the icount trigger must not match.

If neither of these features exist, then single step is doable, but tricky to get right. To single step, the debug stub would execute something like:

```
li    t0, \FcsrIcountCount=4, \FcsrIcountAction=0, \FcsrIcountM=1
csrw  tdata1, t0    /* Write the trigger. */
lw    t0, 8(sp)     /* Restore t0, count decrements to 3 */
lw    sp, 0(sp)     /* Restore sp, count decrements to 2 */
mret                /* Return to program being debugged. count decrements to 1 */
```

There is an additional problem with using to single step. An instruction may cause an exception into a more privileged mode where the trigger is not enabled. The exception handler might address the cause of the exception, and then restart the instruction. Examples of this include page faults, FPU instructions when the FPU is not yet enabled, and interrupts. When a user is single stepping through such code, they will have to step twice to get past the restarted instruction. The first time the exception handler runs, and the second time the instruction actually executes. That is confusing and usually undesirable.

To help users out, debuggers should detect when a single step restarted an instruction, and then step again. This way the users see the expected behavior of stepping over the instruction. Ideally the debugger would notify the user that an exception handler executed the first time.

The debugger should perform this extra step when the PC doesn't change during a regular step.

It is safe to perform an extra step when the PC changes, because every RISC-V instruction either changes the PC or has side effects when repeated, but never both.

To avoid an infinite loop if the exception handler does not address the cause of the exception, the debugger must execute no more than a single extra step. = RISC-V Debug Specification
Version 1.0-
Editors:
Paul Donahue <pdonahue@ventanamicro.com>, Ventana Micro Systems
Tim Newsome <tim@sifive.com>, SiFive, Inc.

RISC-V Debug Specification Version 1.0-

Chapter 9. Preface

Contributors to all versions of the spec in alphabetical order (please contact editors to suggest corrections): Bruce Ableidinger, Krste Asanović, Peter Ashenden, Allen Baum, Mark Beal, Alex Bradbury, Chuanhua Chang, Yen Hao Chen, Zhong-Ho Chen, Monte Dalrymple, Paul Donahue, Vyacheslav Dyachenko, Ernie Edgar, Peter Egold, Marc Gauthier, Markus Goehrle, Robert Golla, John Hauser, Richard Herveille, Yung-ching Hsiao, Po-wei Huang, Scott Johnson, L. J. Madar, Grigorios Magklis, Daniel Mangum, Alexis Marquet, Jan Matyas, Kai Meinhard, Jean-Luc Nagel, Aram Nahidipour, Rishiyur Nikhil, Gajinder Panesar, Deepak Panwar, Antony Pavlov, Klaus Kruse Pedersen, Ken Pettit, Darius Rad, Joe Rahmeh, Josh Scheid, Vedvyas Shanbhogue, Gavin Stark, Ben Staveley, Wesley Terpstra, Tommy Thorn, Megan Wachs, Jan-Willem van de Waerdt, Philipp Wagner, Stefan Wallentowitz, Ray Van De Walker, Andrew Waterman, Thomas Wicki, Andy Wright, Bryan Wyatt, and Florian Zaruba.

Chapter 10. Introduction

When a design progresses from simulation to hardware implementation, a user's control and understanding of the system's current state drops dramatically. To help bring up and debug low level software and hardware, it is critical to have good debugging support built into the hardware. When a robust OS is running on a core, software can handle many debugging tasks. However, in many scenarios, hardware support is essential.

This document outlines a standard architecture for debug support on RISC-V hardware platforms. This architecture allows a variety of implementations and tradeoffs, which is complementary to the wide range of RISC-V implementations. At the same time, this specification defines common interfaces to allow debugging tools and components to target a variety of hardware platforms based on the RISC-V ISA.

System designers may choose to add additional hardware debug support, but this specification defines a standard interface for common functionality.

10.1. Terminology

An advanced feature for advanced users. Most users will not be able to take advantage of it.

Atomic Memory Operation.

JTAG instruction that selects a single bit data register, also called BYPASS.

A RISC-V core, or other part of a hardware platform. Typically all components will be connected to a single system bus.

Control and Status Register.

Debug Module (see Chapter #chap:dm[4]).

Debug Module Interface (see Section 4.1).

JTAG Data Register.

Debug Transport Module (see Section 7).

Debug XLEN, which is the widest XLEN a hart supports, ignoring the current value of $|MXL|$ in `misa`.

An essential feature must be present in order for debug to work correctly.

General Purpose Register.

A single system consisting of one or more *components*.

A hardware thread in a RISC-V core.

32-bit Identification CODE, and a JTAG instruction that returns the IDCODE value.

JTAG Instruction Register.

Refers to work done by IEEE's Joint Test Action Group, described in IEEE 1149.1.

A legacy feature should only be implemented to support legacy hardware that is present in a system.

A subset of the full Debug Specification that allows for very small implementations. See Chapter #chap:dm[4].

Naturally Aligned Power-Of-Two.

Non-Maskable Interrupt.

An address that is directly usable on the system bus.

A recommended feature is not required for debug to work correctly, but it is so useful that it should not be omitted without good reason.

System Bus Access (see Section 4.10).

A specialized feature, that only makes sense in the context of some specific hardware.

Test Access Port, defined in IEEE 1149.1.

Trigger Module (see Section #sec:trigger[6]).

An address as a hart sees it. If the hart is using address translation this may be different from the physical address. If there is no translation then it will be the same.

The exception program counter CSR (e.g. `mepc`) that is appropriate for the mode being trapped to.

10.2. Context

This specification attempts to support all RISC-V ISA extensions that have, roughly, been ratified through the first half of 2023. In particular, though, this specification specifically addresses features in the following extensions:

1. A
2. C
3. D
4. F
5. H
6. Sm1p13
7. Ss1p13
8. Smstateen
9. V
10. Zawrs
11. Zcmp
12. Zicbom
13. Zicboz
14. Zicbop

10.2.1. Versions

Version 0.13 of this document was ratified by the RISC-V Foundation's board. Versions 0.13.x are bug fix releases to that ratified specification.

Version 0.14 was a working version that was never officially ratified.

Version 1.0 is almost entirely forwards and backwards compatible with Version 0.13.

10.2.1.1. Bugfixes from 0.13 to 1.0

Changes that fix a bug in the spec:

1. Fix order of operations described in . #392
2. Resume ack is set after resume, in Section 4.5. #400
3. applies to . #402
4. only applies when action=0. #411
5. does not affect Argument Width. #420
6. Clarify that harts halt out of reset if . #419

10.2.1.2. Incompatible Changes from 0.13 to 1.0

Changes that are not backwards-compatible. Debuggers or hardware implementations that implement 0.13 will have to change something in order to implement 1.0:

1. Make haltsum0 optional if there is only one hart. #505
2. System bus autoincrement only happens if an access actually takes place. () #507
3. Bump to 3. #512
4. Require debugger to poll after lowering it. #566
5. Add to . #574
6. When a selected trigger is disabled, and can be written with any value supported by any of the types this trigger supports. #721
7. fields only apply to breakpoint traps, not any trap. #723
8. If is greater than 0, then (previously called .|hit|) now contains 0 when a trigger fires more than one instruction after the instruction that matched. (This information is now reflected in .) #795
9. If is greater than 0, then bit 20 of is no longer used for timing information. (Previously the bit was called .|$timing$|.) #807
10. If is greater than 0, then the encodings of for sizes greater than 64 bit have changed. #807

10.2.1.3. Minor Changes from 0.13 to 1.0

Changes that slightly modify defined behavior. Technically backwards incompatible, but unlikely to be noticeable:

1. only applies to hart-local counters. #405
2. may be invalid when . #414

3. Address triggers () may fire on any accessed address. #421
4. All trigger registers (Section [csrTrigger]) are optional. #431
5. When extending IR, still is all ones. #437
6. and are WARL. #458
7. NMIs are disabled by . #465
8. R/W1C fields should be cleared by writing every bit high. #472
9. Specify trigger priorities in Table #tab:priority relative to exceptions. #478
10. Time may pass before becomes high. #500
11. Clear MPRV when resuming into lower privilege mode. #503
12. Halt state may not be preserved across reset. #504
13. Hardware should clear trigger action when is cleared and action is 1. #501
14. Change quick access exceptions to halt the target in Section [acQuickaccess]. #585
15. Writing 0 to forces a state where and are writable. #598
16. Solutions to deal with reentrancy in Section #sec:nativetrigger[6.4] prevent triggers from *matching*, not merely *firing*. This primarily affects behavior. #722
17. Attempts to access an unimplemented CSR raise an illegal instruction exception. #791

10.2.1.4. New Features from 0.13 to 1.0

New backwards-compatible feature that did not exist before:

1. Add halt groups and external triggers in Section 4.6. #404
2. Reserve some DMI space for non-standard use. See , and through . #406
3. Reserve trigger values for non-standard use. #417
4. Add bit to . #408 and #709
5. Recommend matching on every accessed address. #449
6. Add resume groups in Section 4.6. #506
7. Add . #536
8. Move , renaming original to , and create . #535
9. Add , deprecating . #538
10. Add hypervisor support: , , , , , , and . #549
11. Optionally make and sticky, controlled by . #520
12. Add to support trigger module external trigger inputs. #543
13. Describe and behavior with atomic instructions. #561
14. Trigger hit bits must be set on fire, may be set on match. #593
15. Add and to and . #588
16. Allow debugger to request harts stay alive with keepalive bit in Section [keepalive]. #592
17. Add to allow a debugger to determine when ndmreset is complete. #594
18. Add to support triggers from an interrupt controller. #599

10.2.1.5. Incompatible Changes During 1.0 Stable

Backwards-incompatible changes between two versions that are both called 1.0 stable.

1. was moved from to , and is now subject to the mode bits in that trigger.
2. #728 introduced Message Registers, which were later removed in #878.
3. It may not be possible to read the contents of the Program Buffer using the `progbuf` registers. #731
4. fields apply to all traps, not just breakpoint traps. This reverts #723. #880

10.3. About This Document

10.3.1. Structure

This document contains two parts. The main part of the document is the specification, which is given in the numbered chapters. The second part of the document is a set of appendices. The information in the appendices is intended to clarify and provide examples, but is not part of the actual specification.

10.3.2. ISA vs. non-ISA

This specification contains both ISA and non-ISA parts. The ISA parts define self-contained ISA extensions. The other parts of the document describe the non-ISA external debug extension. Chapters whose contents are solely one or the other are labeled as such in their title. Chapters without such a label apply to both ISA and non-ISA.

10.3.3. Register Definition Format

All register definitions in this document follow the format shown below. A simple graphic shows which fields are in the register. The upper and lower bit indices are shown to the top left and top right of each field. The total number of bits in the field are shown below it.

After the graphic follows a table which for each field lists its name, description, allowed accesses, and reset value. The allowed accesses are listed in Table #tab:access. The reset value is either a constant or ``Preset." The latter means it is an implementation-specific legal value.

Parts of the register which are currently unused are labeled with the number 0. Software must only write 0 to those fields, and ignore their value while reading. Hardware must return 0 when those fields are read, and ignore the value written to them.

> – –
>
> ```
>  +
> This behavior enables us to use those fields later without having to
> increase the values in the version fields.
> ```

Names of registers and their fields are hyperlinks to their definition, and are also listed in the index on page .

|l|L| R & Read-only.

R/W & Read/Write.
R/W1C & Read/Write Ones to Clear. Writing 0 to every bit has no effect. Writing 1 to every bit clears the field. The result of other writes is undefined.
WARZ & Write any, read zero. A debugger may write any value. When read this field returns 0.
W1 & Write-only. Only writing 1 has an effect. When read the returned value should be 0.
WARL & Write any, read legal. A debugger may write any value. If a value is unsupported, the implementation converts the value to one that is supported.

10.4. Background

There are several use cases for dedicated debugging hardware, both in native debug and external debug. Native debug (sometimes called self-hosted debug) refers to debug software running on a RISC-V platform which debugs the same platform. The optional Trigger Module provides features that are useful for native debug. External debug refers to debug software running somewhere else, debugging the RISC-V platform via a debug transport like JTAG. The entire document provides features that are useful for external debug.

This specification addresses the use cases listed below. Implementations can choose not to implement every feature, which means some use cases might not be supported.

- Accessing hardware on a hardware platform without a working CPU. (External debug.)
- Bootstrapping a hardware platform to test, configure, and program components before there is any executable code path in the hardware platform. (External debug.)
- Debugging low-level software in the absence of an OS or other software. (External debug.)
- Debugging issues in the OS itself. (External or native debug.)
- Debugging processes running on an OS. (Native or external debug.)

10.5. Supported Features

The debug interface described in this specification supports the following features:

1. All hart registers (including CSRs) can be read/written.
2. Memory can be accessed either from the hart's point of view, through the system bus directly, or both.
3. RV32, RV64, and future RV128 are all supported.
4. Any hart in the hardware platform can be independently debugged.
5. A debugger can discover almost[1] everything it needs to know itself, without user configuration.
6. Each hart can be debugged from the very first instruction executed.
7. A RISC-V hart can be halted when a software breakpoint instruction is executed.
8. Hardware single-step can execute one instruction at a time.
9. Debug functionality is independent of the debug transport used.
10. The debugger does not need to know anything about the microarchitecture of the harts it is debugging.
11. Arbitrary subsets of harts can be halted and resumed simultaneously. (Optional)
12. Arbitrary instructions can be executed on a halted hart. That means no new debug functionality is

needed when a core has additional or custom instructions or state, as long as there exist programs that can move that state into GPRs. (Optional)

13. Registers can be accessed without halting. (Optional)
14. A running hart can be directed to execute a short sequence of instructions, with little overhead. (Optional)
15. A system bus manager allows memory access without involving any hart. (Optional)
16. A RISC-V hart can be halted when a trigger matches the PC, read/write address/data, or an instruction opcode. (Optional)
17. Harts can be grouped, and harts in the same group will all halt when any of them halts. These groups can also react to or notify external triggers. (Optional)

This document does not suggest a strategy or implementation for hardware test, debugging or error detection techniques. Scan, built-in self test (BIST), etc. are out of scope of this specification, but this specification does not intend to limit their use in RISC-V systems.

It is possible to debug code that uses software threads, but there is no special debug support for it.

[1] Notable exceptions include information about the memory map and peripherals.

Chapter 11. System Overview

Figure #fig:overview[3.1] shows the main components of Debug Support. Blocks shown in dotted lines are optional.

[fig/overview-eps-converted-to] | *fig/overview-eps-converted-to.pdf*

Figure 2. RISC-V Debug System Overview

The user interacts with the Debug Host (e.g. laptop), which is running a debugger (e.g. gdb). The debugger communicates with a Debug Translator (e.g. OpenOCD, which may include a hardware driver) to communicate with Debug Transport Hardware (e.g. Olimex USB-JTAG adapter). The Debug Transport Hardware connects the Debug Host to the hardware platform's Debug Transport Module (DTM). The DTM provides access to one or more Debug Modules (DMs) using the Debug Module Interface (DMI).

Each hart in the hardware platform is controlled by exactly one DM. Harts may be heterogeneous. There is no further limit on the hart-DM mapping, but usually all harts in a single core are controlled by the same DM. In most hardware platforms there will only be one DM that controls all the harts in the hardware platform.

DMs provide run control of their harts in the hardware platform. Abstract commands provide access to GPRs. Additional registers are accessible through abstract commands or by writing programs to the optional Program Buffer.

The Program Buffer allows the debugger to execute arbitrary instructions on a hart. This mechanism can also be used to access memory. An optional system bus access block allows memory accesses without using a RISC-V hart to perform the access.

Each RISC-V hart may implement a Trigger Module. When trigger conditions are met, harts will halt and inform the debug module that they have halted.

Chapter 12. Debug Module (DM) (non-ISA extension)

The Debug Module implements a translation interface between abstract debug operations and their specific implementation. It might support the following operations:

1. Give the debugger necessary information about the implementation. (Required)
2. Allow any individual hart to be halted and resumed. (Required)
3. Provide status on which harts are halted. (Required)
4. Provide abstract read and write access to a halted hart's GPRs. (Required)
5. Provide access to a reset signal that allows debugging from the very first instruction after reset. (Required)
6. Provide a mechanism to allow debugging harts immediately out of reset (regardless of the reset cause). (Optional)
7. Provide abstract access to non-GPR hart registers. (Optional)
8. Provide a Program Buffer to force the hart to execute arbitrary instructions. (Optional)
9. Allow multiple harts to be halted, resumed, and/or reset at the same time. (Optional)
10. Allow memory access from a hart's point of view. (Optional)
11. Allow direct System Bus Access. (Optional)
12. Group harts. When any hart in the group halts, they all halt. (Optional)
13. Respond to external triggers by halting each hart in a configured group. (Optional)
14. Signal an external trigger when a hart in a group halts. (Optional)

In order to be compatible with this specification an implementation must:

1. Implement all the required features listed above.
2. Implement at least one of Program Buffer, System Bus Access, or Abstract Access Memory command mechanisms.
3. Do at least one of:
    a. Implement the Program Buffer.
    b. Implement abstract access to all registers that are visible to software running on the hart including all the registers that are present on the hart and listed in Table #tab:regno.
    c. Implement abstract access to at least all GPRs, , and , and advertise the implementation as conforming to the `Minimal RISC-V Debug Specification 1.0-'', instead of the` RISC-V Debug Specification 1.0-".

A single DM can debug up to 2^{20} harts.

12.1. Debug Module Interface (DMI)

Debug Modules are subordinates on a bus called the Debug Module Interface (DMI). The bus manager is the Debug Transport Module(s). The Debug Module Interface can be a trivial bus with one manager

and one subordinate (see 8.3), or use a more full-featured bus like TileLink or the AMBA Advanced Peripheral Bus. The details are left to the system designer.

The DMI uses between 7 and 32 address bits. Each address points at a single 32-bit register that can be read or written. The bottom of the address space is used for the first (and usually only) DM. Extra space can be used for custom debug devices, other cores, additional DMs, etc. If there are additional DMs on this DMI, the base address of the next DM in the DMI address space is given in .

The Debug Module is controlled via register accesses to its DMI address space.

12.2. Reset Control

There are two methods that allow a debugger to reset harts. resets all the harts in the hardware platform, as well as all other parts of the hardware platform except for the Debug Modules, Debug Transport Modules, and Debug Module Interface. Exactly what is affected by this reset is implementation dependent, but it must be possible to debug programs from the first instruction executed. resets all the currently selected harts. In this case an implementation may reset more harts than just the ones that are selected. The debugger can discover which other harts are reset (if any) by selecting them and checking and .

To perform either of these resets, the debugger first asserts the bit, and then clears it. The actual reset may start as soon as the bit is asserted, but may start an arbitrarily long time after the bit is deasserted. The reset itself may also take an arbitrarily long time. While the reset is on-going, harts are either in the running state, indicating it's possible to perform some abstract commands during this time, or in the unavailable state, indicating it's not possible to perform any abstract commands during this time. Once a hart's reset is complete, `havereset` becomes set. When a hart comes out of reset and or |hyperref [*resethaltreq*{resethaltreq}|$] are set, the hart will immediately enter Debug Mode (halted state). Otherwise, if the hart was initially running it will execute normally (running state) and if the hart was initially halted it should now be running but may be halted.

> — —
>
> ```
>  +
> There is no general, reliable way for the debugger to know when reset
> has actually begun.
> ```

The Debug Module's own state and registers should only be reset at power-up and while in is 0. If there is another mechanism to reset the DM, this mechanism must also reset all the harts accessible to the DM.

Due to clock and power domain crossing issues, it might not be possible to perform arbitrary DMI accesses across hardware platform reset. While or any external reset is asserted, the only supported DM operations are reading and writing . The behavior of other accesses is undefined.

When harts have been reset, they must set a sticky `havereset` state bit. The conceptual `havereset` state bits can be read for selected harts in and in . These bits must be set regardless of the cause of the reset. The `havereset` bits for the selected harts can be cleared by writing 1 to in . The `havereset` bits might or might not be cleared when is low.

12.3. Selecting Harts

Up to 2^{20} harts can be connected to a single DM. Commands issued to the DM only apply to the currently selected harts.

To enumerate all the harts, a debugger must first determine `HARTSELLEN` by writing all ones to |hyperref [*hartsel*{hartsel}|$](assuming the maximum size) and reading back the value to see which bits were actually set. Then it selects each hart starting from 0 until either in is 1, or the highest index (depending on `HARTSELLEN`) is reached.

The debugger can discover the mapping between hart indices and `mhartid` by using the interface to read `mhartid`, or by reading the hardware platform's configuration structure.

12.3.1. Selecting a Single Hart

All debug modules must support selecting a single hart. The debugger can select a hart by writing its index to |hyperref [*hartsel*{hartsel}|$]. Hart indexes start at 0 and are contiguous until the final index.

12.3.2. Selecting Multiple Harts

Debug Modules may implement a Hart Array Mask register to allow selecting multiple harts at once. The nth bit in the Hart Array Mask register applies to the hart with index n. If the bit is 1 then the hart is selected. Usually a DM will have a Hart Array Mask register exactly wide enough to select all the harts it supports, but it's allowed to tie any of these bits to 0.

The debugger can set bits in the hart array mask register using and , then apply actions to all selected harts by setting . If this feature is supported, multiple harts can be halted, resumed, and reset simultaneously. The state of the hart array mask register is not affected by setting or clearing .

Execution of Abstract Commands ignores this mechanism and only applies to the hart selected by |hyperref [*hartsel*{hartsel}|$].

12.4. Hart DM States

Every hart that can be selected is in exactly one of the following four DM states: non-existent, unavailable, running, or halted. Which state the selected harts are in is reflected by , , , , , , , and .

Harts are nonexistent if they will never be part of this hardware platform, no matter how long a user waits. E.g. in a simple single-hart hardware platform only one hart exists, and all others are nonexistent. Debuggers may assume that a hardware platform has no harts with indexes higher than the first nonexistent one.

Harts are unavailable if they might exist/become available at a later time, or if there are other harts with higher indexes than this one. Harts may be unavailable for a variety of reasons including being reset, temporarily powered down, and not being plugged into the hardware platform. That means harts might become available or unavailable at any time, although these events should be rare in hardware platforms built to be easily debugged. There are no guarantees about the state of the hart when it becomes available.

Hardware platforms with very large number of harts may permanently disable some during manufacturing, leaving holes in the otherwise continuous hart index space. In order to let the debugger discover all harts, they must show up as unavailable even if there is no chance of them ever

becoming available.

Harts are running when they are executing normally, as if no debugger was attached. This includes being in a low power mode or waiting for an interrupt, as long as a halt request will result in the hart being halted.

Harts are halted when they are in Debug Mode, only performing tasks on behalf of the debugger.

Which states a hart that is reset goes through is implementation dependent. Harts may be unavailable while reset is asserted, and some time after reset is deasserted. They might transition to running for some time after reset is deasserted. Finally they end up either running or halted, depending on and |hyperref [*resethaltreq*{resethaltreq}|$].

12.5. Run Control

For every hart, the Debug Module tracks 4 conceptual bits of state: halt request, resume ack, halt-on-reset request, and hart reset. (The hart reset and halt-on-reset request bits are optional.) These 4 bits reset to 0, except for resume ack, which may reset to either 0 or 1. The DM receives halted, running, and havereset signals from each hart. The debugger can observe the state of resume ack in and , and the state of halted, running, and havereset signals in , , , , , and . The state of the other bits cannot be observed directly.

When a debugger writes 1 to , each selected hart's halt request bit is set. When a running hart, or a hart just coming out of reset, sees its halt request bit high, it responds by halting, deasserting its running signal, and asserting its halted signal. Halted harts ignore their halt request bit.

When a debugger writes 1 to , each selected hart's resume ack bit is cleared and each selected, halted hart is sent a resume request. Harts respond by resuming, clearing their halted signal, and asserting their running signal. At the end of this process the resume ack bit is set. These status signals of all selected harts are reflected in , , , and . Resume requests are ignored by running harts.

When halt or resume is requested, a hart must respond in less than one second, unless it is unavailable. (How this is implemented is not further specified. A few clock cycles will be a more typical latency).

The DM can implement optional halt-on-reset bits for each hart, which it indicates by setting to 1. This means the DM implements the and bits. Writing 1 to sets the halt-on-reset request bit for each selected hart. When a hart's halt-on-reset request bit is set, the hart will immediately enter debug mode on the next deassertion of its reset. This is true regardless of the reset's cause. The hart's halt-on-reset request bit remains set until cleared by the debugger writing 1 to while the hart is selected, or by DM reset.

If the DM is reset while a hart is halted, it is unspecified whether that hart resumes. Debuggers should use to explicitly resume harts before clearing and disconnecting.

12.6. Halt Groups, Resume Groups, and External Triggers

An optional feature allows a debugger to place harts into two kinds of groups: halt groups and resume groups. It is also possible to add external triggers to a halt and resume groups. At any given time, each hart and each trigger is a member of exactly one halt group and exactly one resume group.

In both halt and resume groups, group 0 is special. Harts in group 0 halt/resume as if groups aren't implemented at all.

When any hart in a halt group halts:

1. That hart halts normally, with reflecting the original cause of the halt.
2. All the other harts in the halt group that are running will quickly halt. for those harts should be set to 6, but may be set to 3. Other harts in the halt group that are halted but have started the process of resuming must also quickly become halted, even if they do resume briefly.
3. Any external triggers in that group are notified.

Adding a hart to a halt group does not automatically halt that hart, even if other harts in the group are already halted.

When an external trigger that's a member of the halt group fires:

1. All the harts in the halt group that are running will quickly halt. for those harts should be set to 6, but may be set to 3. Other harts in the halt group that are halted but have started the process of resuming must also quickly become halted, even if they do resume briefly.

When any hart in a resume group resumes:

1. All the other harts in that group that are halted will quickly resume as soon as any currently executing abstract commands have completed. Each hart in the group sets its resume ack bit as soon as it has resumed. Harts that are in the process of halting should complete that process and stay halted.
2. Any external triggers in that group are notified.

Adding a hart to a resume group does not automatically resume that hart, even if other harts in the group are currently running.

When an external trigger that's a member of the resume group fires:

1. All the harts in that group that are halted will quickly resume as soon as any currently executing abstract commands have completed. Each hart in the group sets its resume ack bit as soon as it has resumed. Harts that are in the process of halting should complete that process and stay halted.

External triggers are abstract concepts that can signal the DM and/or receive signals from the DM. This configuration is done through , where external triggers are referred to by a number. Commonly, external triggers are capable of sending a signal from the hardware platform into the DM, as well as receiving a signal from the DM to take their own action on. It is also allowable for an external trigger to be input-only or output-only. By convention external triggers 0–7 are bidirectional, triggers 8–11 are input-only, and triggers 12–15 are output-only but this is not required.

> — —
>
> ```
>  +
> External triggers could be used to implement near simultaneous
> halting/resuming of all cores in a hardware platform, when not all cores
> are RISC-V cores.
> ```

When the DM is reset, all harts must be placed in the lowest-numbered halt and resume groups that they can be in. (This will usually be group 0.)

Some designs may choose to hardcode hart groups to a group other than group 0, meaning it is never possible to halt or resume just a single hart. This is explicitly allowed. In that case it must be possible to discover the groups by using even if it's not possible to change the configuration.

12.7. Abstract Commands

The DM supports a set of abstract commands, most of which are optional. Depending on the implementation, the debugger may be able to perform some abstract commands even when the selected hart is not halted. Debuggers can only determine which abstract commands are supported by a given hart in a given state (running, halted, or held in reset) by attempting them and then looking at in to see if they were successful. Commands may be supported with some options set, but not with other options set. If a command has unsupported options set or if bits that are defined as 0 aren't 0, then the DM must set to 2 (not supported).

> – –
>
> ```
>  +
> Example: Every DM must support the Access Register command, but might
> not support accessing CSRs. If the debugger requests to read a CSR in
> that case, the command will return ``not supported.''
> ```

Debuggers execute abstract commands by writing them to . They can determine whether an abstract command is complete by reading in . If the debugger starts a new command while is set, becomes 1 (busy), the currently executing command still gets to run to completion, but any error generated by the currently executing command is lost. After completion, indicates whether the command was successful or not. Commands may fail because a hart is not halted, not running, unavailable, or because they encounter an error during execution.

If the command takes arguments, the debugger must write them to the `data` registers before writing to . If a command returns results, the Debug Module must ensure they are placed in the `data` registers before is cleared. Which `data` registers are used for the arguments is described in Table #tab:datareg. In all cases the least-significant word is placed in the lowest-numbered `data` register. The argument width depends on the command being executed, and is DXLEN where not explicitly specified.

|r|l|l|l| Argument Width & arg0/return value & arg1 & arg2
& & `data1` & `data2`
& , `data1` & `data2`, `data3` & `data4`, `data5`
& –`data3` & `data4`–`data7` & `data8`–`data11`

> – –
>
> ```
>  +
> The Abstract Command interface is designed to allow a debugger to write
> commands as fast as possible, and then later check whether they
> completed without error. In the common case the debugger will be much
> ```

```
slower than the target and commands succeed, which allows for maximum
throughput. If there is a failure, the interface ensures that no
commands execute after the failing one. To discover which command
failed, the debugger has to look at the state of the DM (e.g. contents
of ) or hart (e.g. contents of a register modified by a Program Buffer
program) to determine which one failed.
```

Before starting an abstract command, a debugger must ensure that , , and are all 0.

While an abstract command is executing (in is high), a debugger must not change |hyperref [*hartsel*{hartsel}|$], and must not write 1 to , , , , or .

If an abstract command does not complete in the expected time and appears to be hung, the debugger can try to reset the hart (using or). If that doesn't clear , then it can try resetting the Debug Module (using).

If an abstract command is started while the selected hart is unavailable or if a hart becomes unavailable while executing an abstract command, then the Debug Module may terminate the abstract command, setting low, and to 4 (halt/resume). Alternatively, the command could just appear to be hung (never goes low).

12.7.1. Abstract Command Listing

This section describes each of the different abstract commands and how their fields should be interpreted when they are written to .

Each abstract command is a 32-bit value. The top 8 bits contain which determines the kind of command. Table #tab:cmdtype lists all commands.

|r|l|l|l| & Command & Page
& Access Register Command &
& Quick Access &
& Access Memory Command &

12.8. Program Buffer

To support executing arbitrary instructions on a halted hart, a Debug Module can include a Program Buffer that a debugger can write small programs to. DMs that support all necessary functionality using abstract commands only may choose to omit the Program Buffer.

A debugger can write a small program to the Program Buffer, and then execute it exactly once with the Access Register Abstract Command, setting the bit in . The debugger can write whatever program it likes (including jumps out of the Program Buffer), but the program must end with `ebreak` or `c.ebreak`. An implementation may support an implicit `ebreak` that is executed when a hart runs off the end of the Program Buffer. This is indicated by . With this feature, a Program Buffer of just 2 32-bit words can offer efficient debugging.

While these programs are executed, the hart does not leave Debug Mode (see Section 5.1). If an exception is encountered during execution of the Program Buffer, no more instructions are executed, the hart remains in Debug Mode, and is set to 3 (`exception error`). If the debugger executes a program that doesn't terminate with an `ebreak` instruction, the hart will remain in Debug Mode and the debugger will lose control of the hart.

If is 1 then the following apply:

1. must be 1.
2. If the debugger writes a compressed instruction into the Program Buffer, it must be placed into the lower 16 bits and accompanied by a compressed `nop` in the upper 16 bits.

> _ _
>
> ```
>  +
> This requirement on the debugger for the case of equal to 1 is to
> accommodate hardware designs that prefer to stuff instructions directly
> into the pipeline when halted, instead of having the Program Buffer
> exist in the address space somewhere.
> ```

The Program Buffer may be implemented as RAM which is accessible to the hart. A debugger can determine if this is the case by executing small programs that attempt to write and read back relative to `pc` while executing from the Program Buffer. If so, the debugger has more flexibility in what it can do with the program buffer.

12.9. Overview of Hart Debug States

Figure #fig:abstract_sm[4.1] shows a conceptual view of the states passed through by a hart during run/halt debugging as influenced by the different fields of , , , and .

[fig/abstract_commands] | *fig/abstract_commands.pdf*

Figure 3. Run/Halt Debug State Machine for single-hart hardware platforms. As only a small amount of state is visible to the debugger, the states and transitions are conceptual.

12.10. System Bus Access

A debugger can access memory from a hart's point of view using a Program Buffer or the Abstract Access Memory command. (Both these features are optional.) A Debug Module may also include a System Bus Access block to provide memory access without involving a hart, regardless of whether Program Buffer is implemented. The System Bus Access block uses physical addresses.

The System Bus Access block may support 8-, 16-, 32-, 64-, and 128-bit accesses. Table #tab:sbdatabits shows which bits in `sbdata` are used for each access size.

|r|l| Access Size & Data Bits
& bits 7:0
& bits 15:0
&
& ,
& , , ,

Depending on the microarchitecture, data accessed through System Bus Access might not always be coherent with that observed by each hart. It is up to the debugger to enforce coherency if the implementation does not. This specification does not define a standard way to do this. Possibilities may include writing to special memory-mapped locations, or executing special instructions via the

Program Buffer.

> — —
>
> ```
>  +
> Implementing a System Bus Access block has several benefits even when a
> Debug Module also implements a Program Buffer. First, it is possible to
> access memory in a running system with minimal impact. Second, it may
> improve performance when accessing memory. Third, it may provide access
> to devices that a hart does not have access to.
> ```

12.11. Minimally Intrusive Debugging

Depending on the task it is performing, some harts can only be halted very briefly. There are several mechanisms that allow accessing resources in such a running system with a minimal impact on the running hart.

First, an implementation may allow some abstract commands to execute without halting the hart.

Second, the Quick Access abstract command can be used to halt a hart, quickly execute the contents of the Program Buffer, and let the hart run again. Combined with instructions that allow Program Buffer code to access the `data` registers, as described in , this can be used to quickly perform a memory or register access. For some hardware platforms this will be too intrusive, but many hardware platforms that can't be halted can bear an occasional hiccup of a hundred or less cycles.

Third, if the System Bus Access block is implemented, it can be used while a hart is running to access system memory.

12.12. Security

To protect intellectual property it may be desirable to lock access to the Debug Module. To allow access during a manufacturing process and not afterwards, a reasonable solution could be to add a fuse bit to the Debug Module that can be used to be permanently disable it. Since this is technology specific, it is not further addressed in this spec.

Another option is to allow the DM to be unlocked only by users who have an access key. Between , , and arbitrarily complex authentication mechanism can be supported. When is clear, the DM must not interact with the rest of the hardware platform, nor expose details about the harts connected to the DM. All DM registers should read 0, while writes should be ignored, with the following mandatory exceptions:

1. in is readable.
2. in is readable.
3. in is readable.
4. in is readable and writable.
5. is readable and writable.

Implementations where it's not possible to unlock the DM by using should not implement that

register.

12.13. Version Detection

To detect the version of the Debug Module with a minimum of side effects, use the following procedure:

1. Read .
2. If is 0 or is 1:
   a. Write , preserving , , , and from the value that was read, setting , and clearing all the other bits.
   b. Read until is high.
3. Read , which contains .

If it was necessary to clear , this might have the following unavoidable side effects:

1. is cleared, potentially preventing a halt request made by a previous debugger from taking effect.
2. is cleared, potentially preventing a resume request made by a previous debugger from taking effect.
3. is deasserted, releasing the hardware platform from reset if a previous debugger had set it.
4. is asserted, releasing the DM from reset. This in itself is not observable by any harts.

This procedure is guaranteed to work in future versions of this spec. The meaning of the bits where , , , and currently reside might change, but preserving them will have no side effects. Clearing the bits of not explicitly mentioned here will have no side effects beyond the ones mentioned above.

12.14. Debug Module Registers

The registers described in this section are accessed over the DMI bus. Each DM has a base address (which is 0 for the first DM). The register addresses below are offsets from this base address.

Debug Module DMI Registers that are unimplemented or not mentioned in the table below return 0 when read. Writing them has no effect.

Address	Name	Page
Continued on next page		
0x04	Abstract Data 0 (`data0`)	
0x05	Abstract Data 1 (`data1`)	
0x06	Abstract Data 2 (`data2`)	
0x07	Abstract Data 3 (`data3`)	
0x08	Abstract Data 4 (`data4`)	
0x09	Abstract Data 5 (`data5`)	
0x0a	Abstract Data 6 (`data6`)	
0x0b	Abstract Data 7 (`data7`)	
0x0c	Abstract Data 8 (`data8`)	

Address	Name	Page
0x0d	Abstract Data 9 (`data9`)	
0x0e	Abstract Data 10 (`data10`)	
0x0f	Abstract Data 11 (`data11`)	
0x10	Debug Module Control (`dmcontrol`)	
0x11	Debug Module Status (`dmstatus`)	
0x12	Hart Info (`hartinfo`)	
0x13	Halt Summary 1 (`haltsum1`)	
0x14	Hart Array Window Select (`hawindowsel`)	
0x15	Hart Array Window (`hawindow`)	
0x16	Abstract Control and Status (`abstractcs`)	
0x17	Abstract Command (`command`)	
0x18	Abstract Command Autoexec (`abstractauto`)	
0x19	Configuration Structure Pointer 0 (`confstrptr0`)	
0x1a	Configuration Structure Pointer 1 (`confstrptr1`)	
0x1b	Configuration Structure Pointer 2 (`confstrptr2`)	
0x1c	Configuration Structure Pointer 3 (`confstrptr3`)	
0x1d	Next Debug Module (`nextdm`)	
0x1f	Custom Features (`custom`)	
0x20	Program Buffer 0 (`progbuf0`)	
0x21	Program Buffer 1 (`progbuf1`)	
0x22	Program Buffer 2 (`progbuf2`)	
0x23	Program Buffer 3 (`progbuf3`)	
0x24	Program Buffer 4 (`progbuf4`)	
0x25	Program Buffer 5 (`progbuf5`)	
0x26	Program Buffer 6 (`progbuf6`)	
0x27	Program Buffer 7 (`progbuf7`)	
0x28	Program Buffer 8 (`progbuf8`)	
0x29	Program Buffer 9 (`progbuf9`)	
0x2a	Program Buffer 10 (`progbuf10`)	
0x2b	Program Buffer 11 (`progbuf11`)	
0x2c	Program Buffer 12 (`progbuf12`)	
0x2d	Program Buffer 13 (`progbuf13`)	

Address	Name	Page
0x2e	Program Buffer 14 (progbuf14)	
0x2f	Program Buffer 15 (progbuf15)	
0x30	Authentication Data (authdata)	
0x32	Debug Module Control and Status 2 (dmcs2)	
0x34	Halt Summary 2 (haltsum2)	
0x35	Halt Summary 3 (haltsum3)	
0x37	System Bus Address 127:96 (sbaddress3)	
0x38	System Bus Access Control and Status (sbcs)	
0x39	System Bus Address 31:0 (sbaddress0)	
0x3a	System Bus Address 63:32 (sbaddress1)	
0x3b	System Bus Address 95:64 (sbaddress2)	
0x3c	System Bus Data 31:0 (sbdata0)	
0x3d	System Bus Data 63:32 (sbdata1)	
0x3e	System Bus Data 95:64 (sbdata2)	
0x3f	System Bus Data 127:96 (sbdata3)	
0x40	Halt Summary 0 (haltsum0)	
0x70	Custom Features 0 (custom0)	
0x71	Custom Features 1 (custom1)	
0x72	Custom Features 2 (custom2)	
0x73	Custom Features 3 (custom3)	
0x74	Custom Features 4 (custom4)	
0x75	Custom Features 5 (custom5)	
0x76	Custom Features 6 (custom6)	
0x77	Custom Features 7 (custom7)	
0x78	Custom Features 8 (custom8)	
0x79	Custom Features 9 (custom9)	
0x7a	Custom Features 10 (custom10)	
0x7b	Custom Features 11 (custom11)	
0x7c	Custom Features 12 (custom12)	
0x7d	Custom Features 13 (custom13)	
0x7e	Custom Features 14 (custom14)	
0x7f	Custom Features 15 (custom15)	

Table 7. Debug Module Debug Bus Registers

12.14.1. Debug Module Status (`dmstatus`, at 0x11)

[dmDmstatus]# This register reports status for the overall Debug Module as well as the currently selected harts, as defined in . Its address will not change in the future, because it contains .

This entire register is read-only.

p3.5 exp3.5 exp7.5 exp7.5 exp6.5 exp6.5 exp4.5 exp4.5 exp2.4 exp2.4 exp6.0 exp6.0 ex 31 & & & & & 21 & &
& & & & &
& & & & &

p6.0 exp6.0 exp6.0 exp6.0 exp6.0 exp6.0 exp7.0 exp7.0 exp7.0 exp7.0 exp5.0 exp5.0 ex & & & & & & & & & &
& & & & &

p5.0 exp5.0 exp5.0 exp5.0 exp5.0 exp5.0 exp4.5 exp4.5 exp4.5 exp4.5 exp6.5 exp6.5 ex & & & & & & & & & &
& & & & &

p4.0 exp4.0 exp7.5 exp7.5 exp7.5 exp7.5 exp3.5 exp3.5 ex & & & 3 & & & &
& & &

|l|p0.5|c|l| Field & Description & Access & Reset

[dmDmstatusNdmresetpending]# |ndmresetpending| &

0 (false): Unimplemented, or is zero and no ndmreset is currently in progress.

1 (true): is currently nonzero, or there is an ndmreset in progress. & R & -
[dmDmstatusStickyunavail]# |stickyunavail| &

0 (current): The per-hart `unavail` bits reflect the current state of the hart.

1 (sticky): The per-hart `unavail` bits are sticky. Once they are set, they will not clear until the debugger acknowledges them using . & R & Preset
[dmDmstatusImpebreak]# |impebreak| & If 1, then there is an implicit `ebreak` instruction at the non-existent word immediately after the Program Buffer. This saves the debugger from having to write the `ebreak` itself, and allows the Program Buffer to be one word smaller.

This must be 1 when is 1. & R & Preset
|allhavereset| & This field is 1 when all currently selected harts have been reset and reset has not been acknowledged for any of them. & R & -
[dmDmstatusAnyhavereset] |anyhavereset| & This field is 1 when at least one currently selected hart has been reset and reset has not been acknowledged for that hart. & R & -
|allresumeack| & This field is 1 when all currently selected harts have their resume ack bit set. & R & -
[#dmDmstatusAnyresumeack] |anyresumeack| & This field is 1 when any currently selected hart has its resume ack bit set. & R & -
|allnonexistent| & This field is 1 when all currently selected harts do not exist in this hardware platform. & R & -
[#dmDmstatusAnynonexistent] |anynonexistent| & This field is 1 when any currently selected hart

does not exist in this hardware platform. & R & -
|allunavail| & This field is 1 when all currently selected harts are unavailable, or (if is 1) were unavailable without that being acknowledged. & R & -
[#dmDmstatusAnyunavail] |anyunavail| & This field is 1 when any currently selected hart is unavailable, or (if is 1) was unavailable without that being acknowledged. & R & -
|allrunning| & This field is 1 when all currently selected harts are running. & R & -
[#dmDmstatusAnyrunning] |anyrunning| & This field is 1 when any currently selected hart is running. & R & -
|allhalted| & This field is 1 when all currently selected harts are halted. & R & -
[#dmDmstatusAnyhalted] |anyhalted| & This field is 1 when any currently selected hart is halted. & R & -
[#dmDmstatusAuthenticated]# |authenticated| &

0 (false): Authentication is required before using the DM.

1 (true): The authentication check has passed.

On components that don't implement authentication, this bit must be preset as 1. & R & Preset
[dmDmstatusAuthbusy]# |authbusy| &

0 (ready): The authentication module is ready to process the next read/write to .

1 (busy): The authentication module is busy. Accessing results in unspecified behavior.

only becomes set in immediate response to an access to . & R & 0
|hasresethaltreq| & 1 if this Debug Module supports halt-on-reset functionality controllable by the and bits. 0 otherwise. & R & Preset
[#dmDmstatusConfstrptrvalid] |confstrptrvalid| &

0 (invalid): –hold information which is not relevant to the configuration structure.

1 (valid): –hold the address of the configuration structure. & R & Preset
[dmDmstatusVersion]# |version| &

0 (none): There is no Debug Module present.

1 (0.11): There is a Debug Module and it conforms to version 0.11 of this specification.

2 (0.13): There is a Debug Module and it conforms to version 0.13 of this specification.

3 (1.0): There is a Debug Module and it conforms to version 1.0 of this specification.

15 (custom): There is a Debug Module but it does not conform to any available version of this spec. & R & 3

12.14.2. Debug Module Control (`dmcontrol`, at 0x10)

[dmDmcontrol]# This register controls the overall Debug Module as well as the currently selected harts, as defined in .

[hartsel]# Throughout this document we refer to |hyperref [*hartsel*{hartsel}|$], which is combined with . While the spec allows for 20 |hyperref [*hartsel*{hartsel}|$] bits, an implementation may choose to implement fewer than that. The actual width of |hyperref [*hartsel*{hartsel}|$] is called `HARTSELLEN`. It must be at least 0 and at most 20. A debugger should discover `HARTSELLEN` by writing all ones to

|hyperref [*hartsel*{hartsel}|$](assuming the maximum size) and reading back the value to see which bits were actually set. Debuggers must not change |hyperref [*hartsel*{hartsel}|$] while an abstract command is executing.

> – –
>
> ```
>  +
> There are separate and bits so that it is possible to write without
> changing the halt-on-reset request bit for each selected hart, when not
> all selected harts have the same configuration.
> ```

On any given write, a debugger may only write 1 to at most one of the following bits: , , , , and . The others must be written 0.

is an optional internal bit of per-hart state that cannot be read, but can be written with and .

[keepalive]# |hyperref [*keepalive*{keepalive}|$] is an optional internal bit of per-hart state. When it is set, it suggests that the hardware should attempt to keep the hart available for the debugger, e.g. by keeping it from entering a low-power state once powered on. Even if the bit is implemented, hardware might not be able to keep a hart available. The bit is written through and .

For forward compatibility, will always be readable when bit 1 () is 0 and bit 0 () is 1.

p3.5 exp3.5 exp4.5 exp4.5 exp4.5 exp4.5 exp6.0 exp6.0 exp5.0 exp5.0 ex & & & &
& & & &
& & & &

p2.5 exp2.5 exp5.0 exp5.0 exp6.7 exp3.3 exp6.0 exp6.0 exp6.0 exp6.0 ex & 25 & & 15 & & &
& & & &
& & & &

p7.5 exp7.5 exp7.5 exp7.5 exp4.0 exp4.0 exp4.0 exp4.0 ex & & &
& & &
& & &

|l|p0.5|c|l| Field & Description & Access & Reset

[dmDmcontrolHaltreq]# |haltreq| & Writing 0 clears the halt request bit for all currently selected harts. This may cancel outstanding halt requests for those harts.

Writing 1 sets the halt request bit for all currently selected harts. Running harts will halt whenever their halt request bit is set.

Writes apply to the new value of |hyperref [*hartsel*{hartsel}|$] and . & WARZ & -
[dmDmcontrolResumereq]# |resumereq| & Writing 1 causes the currently selected harts to resume once, if they are halted when the write occurs. It also clears the resume ack bit for those harts.

is ignored if is set.

Writes apply to the new value of |hyperref [*hartsel*{hartsel}|$] and . & W1 & -
[dmDmcontrolHartreset]# |hartreset| & This optional field writes the reset bit for all the currently

selected harts. To perform a reset the debugger writes 1, and then writes 0 to deassert the reset signal.

While this bit is 1, the debugger must not change which harts are selected.

If this feature is not implemented, the bit always stays 0, so after writing 1 the debugger can read the register back to see if the feature is supported.

Writes apply to the new value of |hyperref [*hartsel*{hartsel}|$] and . & WARL & 0 [dmDmcontrolAckhavereset]# |ackhavereset| &

0 (nop): No effect.

1 (ack): Clears `havereset` for any selected harts.

Writes apply to the new value of |hyperref [*hartsel*{hartsel}|$] and . & W1 & - [dmDmcontrolAckunavail]# |ackunavail| &

0 (nop): No effect.

1 (ack): Clears `unavail` for any selected harts that are currently available.

Writes apply to the new value of |hyperref [*hartsel*{hartsel}|$] and . & W1 & - [dmDmcontrolHasel]# |hasel| & Selects the definition of currently selected harts.

0 (single): There is a single currently selected hart, that is selected by |hyperref [*hartsel*{hartsel}|$].

1 (multiple): There may be multiple currently selected harts — the hart selected by |hyperref [*hartsel*{hartsel}|$], plus those selected by the hart array mask register.

An implementation which does not implement the hart array mask register must tie this field to 0. A debugger which wishes to use the hart array mask register feature should set this bit and read back to see if the functionality is supported. & WARL & 0
|hartsello| & The low 10 bits of |hyperref [*hartsel*{hartsel}|$]: the DM-specific index of the hart to select. This hart is always part of the currently selected harts. & WARL & 0
[dmDmcontrolHartselhi] |hartselhi| & The high 10 bits of |hyperref [*hartsel*{hartsel}|$]: the DM-specific index of the hart to select. This hart is always part of the currently selected harts. & WARL & 0
[#dmDmcontrolSetkeepalive]# |setkeepalive| & This optional field sets |hyperref [*keepalive*{keepalive}|$] for all currently selected harts, unless is simultaneously set to 1.

Writes apply to the new value of |hyperref [*hartsel*{hartsel}|$] and . & W1 & - [dmDmcontrolClrkeepalive]# |clrkeepalive| & This optional field clears |hyperref [*keepalive*{keepalive}|$] for all currently selected harts.

Writes apply to the new value of |hyperref [*hartsel*{hartsel}|$] and . & W1 & - [dmDmcontrolSetresethaltreq]# |setresethaltreq| & This optional field writes the halt-on-reset request bit for all currently selected harts, unless is simultaneously set to 1. When set to 1, each selected hart will halt upon the next deassertion of its reset. The halt-on-reset request bit is not automatically cleared. The debugger must write to to clear it.

Writes apply to the new value of |hyperref [*hartsel*{hartsel}|$] and .

If is 0, this field is not implemented. & W1 & - [dmDmcontrolClrresethaltreq]# |clrresethaltreq| & This optional field clears the halt-on-reset request bit for all currently selected harts.

Writes apply to the new value of |hyperref [*hartsel*{hartsel}|$] and . & W1 & -
|ndmreset| & This bit controls the reset signal from the DM to the rest of the hardware platform. The signal should reset every part of the hardware platform, including every hart, except for the DM and any logic required to access the DM. To perform a hardware platform reset the debugger writes 1, and then writes 0 to deassert the reset. & R/W & 0
[#dmDmcontrolDmactive] |dmactive| & This bit serves as a reset signal for the Debug Module itself. After changing the value of this bit, the debugger must poll until has taken the requested value before performing any action that assumes the requested state change has completed. Hardware may take an arbitrarily long time to complete activation or deactivation and will indicate completion by setting to the requested value.

0 (inactive): The module's state, including authentication mechanism, takes its reset values (the bit is the only bit which can be written to something other than its reset value). Any accesses to the module may fail. Specifically, might not return correct data.

1 (active): The module functions normally.

No other mechanism should exist that may result in resetting the Debug Module after power up.

To place the Debug Module into a known state, a debugger may write 0 to , poll until is observed 0, write 1 to , and poll until is observed 1.

Implementations may pay attention to this bit to further aid debugging, for example by preventing the Debug Module from being power gated while debugging is active. & R/W & 0

12.14.3. Hart Info (`hartinfo`, at 0x12)

[dmHartinfo]# This register gives information about the hart currently selected by |hyperref [*hartsel*{hartsel}|$].

This register is optional. If it is not present it should read all-zero.

If this register is included, the debugger can do more with the Program Buffer by writing programs which explicitly access the `data` and/or `dscratch` registers.

This entire register is read-only.

p4.0 exp4.0 exp4.0 exp4.0 exp2.4 exp2.4 exp5.0 exp5.0 exp4.0 exp4.0 exp8.0 exp4.0 ex 31 & & 23 & & 19 & & & 15 & & 11 &
& & & & &
& & & & &

|l|p0.5|c|l| Field & Description & Access & Reset

|nscratch| & Number of `dscratch` registers available for the debugger to use during program buffer execution, starting from . The debugger can make no assumptions about the contents of these registers between commands. & R & Preset
[#dmHartinfoDataaccess] |dataaccess| &

0 (csr): The `data` registers are shadowed in the hart by CSRs. Each CSR is DXLEN bits in size, and corresponds to a single argument, per Table #tab:datareg.

1 (memory): The `data` registers are shadowed in the hart's memory map. Each register takes up 4 bytes

in the memory map. & R & Preset
[dmHartinfoDatasize]# |datasize| & If is 0: Number of CSRs dedicated to shadowing the `data` registers.

If is 1: Number of 32-bit words in the memory map dedicated to shadowing the `data` registers.

If this value is non-zero, then the tt data registers must be traditional registers and not MRs.

Since there are at most 12 `data` registers, the value in this register must be 12 or smaller. & R & Preset
[dmHartinfoDataaddr]# |dataaddr| & If is 0: The number of the first CSR dedicated to shadowing the `data` registers.

If is 1: Address of RAM where the data registers are shadowed. This address is sign extended giving a range of -2048 to 2047, easily addressed with a load or store using `x0` as the address register. & R & Preset

12.14.4. Hart Array Window Select (`hawindowsel`, at 0x14)

[dmHawindowsel]# This register selects which of the 32-bit portion of the hart array mask register (see Section 4.3.2) is accessible in .

p8.5 exp8.5 exp10.0 exp5.0 ex 31 & & 14 &
&
&

|l|p0.5|c|l| Field & Description & Access & Reset

[dmHawindowselHawindowsel]# |hawindowsel| & The high bits of this field may be tied to 0, depending on how large the array mask register is. E.g. on a hardware platform with 48 harts only bit 0 of this field may actually be writable. & WARL & 0

12.14.5. Hart Array Window (`hawindow`, at 0x15)

[dmHawindow]# This register provides R/W access to a 32-bit portion of the hart array mask register (see Section 4.3.2). The position of the window is determined by . I.e. bit 0 refers to hart RdmHawindowsel * 32, while bit 31 refers to hart RdmHawindowsel * 32+31.

Since some bits in the hart array mask register may be constant 0, some bits in this register may be constant 0, depending on the current value of .

p21.3 exp10.7 ex 31 &

12.14.6. Abstract Control and Status (`abstractcs`, at 0x16)

[dmAbstractcs]# Writing this register while an abstract command is executing causes to become 1 (busy) once the command completes (busy becomes 0).

> — —
>
> +

```
must be at least 1 to support RV32 harts, 2 to support RV64 harts, or 4
to support RV128 harts.
```

p2.4 exp2.4 exp5.5 exp5.5 exp5.5 exp5.5 exp2.4 exp2.4 ex 31 & & 28 & & 23 & &
& & &
& & &

p5.5 exp5.5 exp4.0 exp2.0 exp2.0 exp2.0 exp4.5 exp4.5 ex & 10 & & 7 & & 3 &
& & &
& & &

|l|p0.5|c|l| Field & Description & Access & Reset

|progbufsize| & Size of the Program Buffer, in 32-bit words. Valid sizes are 0 - 16. & R & Preset
[#dmAbstractcsBusy] |busy| &

0 (ready): There is no abstract command currently being executed.

1 (busy): An abstract command is currently being executed.

This bit is set as soon as is written, and is not cleared until that command has completed. & R & 0
[dmAbstractcsRelaxedpriv]# |relaxedpriv| & This optional bit controls whether program buffer and abstract memory accesses are performed with the exact and full set of permission checks that apply based on the current architectural state of the hart performing the access, or with a relaxed set of permission checks (e.g. PMP restrictions are ignored). The details of the latter are implementation-specific.

0 (full checks): Full permission checks apply.

1 (relaxed checks): Relaxed permission checks apply. & WARL & Preset
[dmAbstractcsCmderr]# |cmderr| & Gets set if an abstract command fails. The bits in this field remain set until they are cleared by writing 1 to them. No abstract command is started until the value is reset to 0.

This field only contains a valid value if is 0.

0 (none): No error.

1 (busy): An abstract command was executing while , , or was written, or when one of the `data` or `progbuf` registers was read or written. This status is only written if contains 0.

2 (not supported): The command in is not supported. It may be supported with different options set, but it will not be supported at a later time when the hart or system state are different.

3 (exception): An exception occurred while executing the command (e.g. while executing the Program Buffer).

4 (halt/resume): The abstract command couldn't execute because the hart wasn't in the required state (running/halted), or unavailable.

5 (bus): The abstract command failed due to a bus error (e.g. alignment, access size, or timeout).

6 (reserved): Reserved for future use.

7 (other): The command failed for another reason. & R/W1C & 0
[dmAbstractcsDatacount]# |datacount| & Number of `data` registers that are implemented as part of the abstract command interface. Valid sizes are 1 – 12. & R & Preset

12.14.7. Abstract Command (`command`, at 0x17)

[dmCommand]# Writes to this register cause the corresponding abstract command to be executed.

Writing this register while an abstract command is executing causes to become 1 (busy) once the command completes (busy becomes 0).

If is non-zero, writes to this register are ignored.

> – –
>
> ```
>  +
> inhibits starting a new command to accommodate debuggers that, for
> performance reasons, send several commands to be executed in a row
> without checking in between. They can safely do so and check at the end
> without worrying that one command failed but then a later command (which
> might have depended on the previous one succeeding) passed.
> ```

31	24	23	0
latexmath:[$	cmdtype	$]	
latexmath:[$	control	$]	
8		24	

Field	Description	Access	Reset
Continued on next page			
[dmCommandCmdtype]# |cmdtype|	The type determines the overall functionality of this abstract command.	WARZ	0
|control|	This field is interpreted in a command-specific manner, described for each abstract command.	WARZ	0

12.14.8. Abstract Command Autoexec (`abstractauto`, at 0x18)

[dmAbstractauto]# This register is optional. Including it allows more efficient burst accesses. A debugger can detect whether it is supported by setting bits and reading them back.

If this register is implemented then bits corresponding to implemented progbuf and data registers must be writable. Other bits must be hard-wired to 0.

If this register is written while an abstract command is executing then the write is ignored and becomes 1 (busy) once the command completes (busy becomes 0).

31	16	15	12	11	0
latexmath:[$	autoexecprogbuf	$]		latexmath:[$	0
$]		latexmath:[$	autoexecdata	$]	
16		4		12	

Field	Description	Access	Reset
Continued on next page			
[dmAbstractautoAutoexecprogbuf]# |autoexecprogbuf|	When a bit in this field is 1, read or write accesses to the corresponding `progbuf` word cause the DM to act as if the current value in was written there again after the access to `progbuf` completes.	WARL	0
|autoexecdata|	When a bit in this field is 1, read or write accesses to the corresponding `data` word cause the DM to act as if the current value in was written there again after the access to `data` completes.	WARL	0

12.14.9. Configuration Structure Pointer 0 (`confstrptr0`, at 0x19)

[dmConfstrptrZero]# When is set, reading this register returns bits 31:0 of the configuration structure pointer. Reading the other `confstrptr` registers returns the upper bits of the address.

When system bus access is implemented, this must be an address that can be used with the System Bus Access module. Otherwise, this must be an address that can be used to access the configuration structure from the hart with ID 0.

If is 0, then the `confstrptr` registers hold identifier information which is not further specified in this document.

The configuration structure itself is a data structure of the same format as the data structure pointed to by mconfigptr as described in the Privileged Spec.

This entire register is read-only.

31	0
latexmath:[$	addr
$]	
32	

12.14.10. Configuration Structure Pointer 1 (`confstrptr1`, at 0x1a)

[dmConfstrptrOne]# When is set, reading this register returns bits 63:32 of the configuration structure pointer. See for more details.

This entire register is read-only.

31	0
latexmath:[$	addr
$]	
32	

12.14.11. Configuration Structure Pointer 2 (confstrptr2, at 0x1b)

[dmConfstrptrTwo]# When is set, reading this register returns bits 95:64 of the configuration structure pointer. See for more details.

This entire register is read-only.

31	0
latexmath:[$	addr
$]	
32	

12.14.12. Configuration Structure Pointer 3 (confstrptr3, at 0x1c)

[dmConfstrptrThree]# When is set, reading this register returns bits 127:96 of the configuration structure pointer. See for more details.

This entire register is read-only.

31	0
latexmath:[$	addr
$]	
32	

12.14.13. Next Debug Module (nextdm, at 0x1d)

[dmNextdm]# If there is more than one DM accessible on this DMI, this register contains the base address of the next one in the chain, or 0 if this is the last one in the chain.

This entire register is read-only.

31	0
latexmath:[$	addr
$]	
32	

12.14.14. Abstract Data 0 (data0, at 0x04)

[dmDataZero]# through may be Message Registers, whose behavior is described in Section #sec:mr. These registers may be read or changed by abstract commands. indicates how many of them are implemented, starting at , counting up. Table #tab:datareg shows how abstract commands use these registers.

Accessing these registers while an abstract command is executing causes to be set to 1 (busy) if it is 0.

Attempts to write them while is set does not change their value.

The values in these registers might not be preserved after an abstract command is executed. The only guarantees on their contents are the ones offered by the command in question. If the command fails, no assumptions can be made about the contents of these registers.

31	0
latexmath:[$	data
$]	
32	

12.14.15. Program Buffer 0 (`progbuf0`, at 0x20)

[dmProgbufZero]# through must provide write access to the optional program buffer. It may also be possible for the debugger to read from the program buffer through these registers. If reading is not supported, then all reads return 0.

indicates how many `progbuf` registers are implemented starting at , counting up.

Accessing these registers while an abstract command is executing causes to be set to 1 (busy) if it is 0.

Attempts to write them while is set does not change their value.

31	0
latexmath:[$	data
$]	
32	

12.14.16. Authentication Data (`authdata`, at 0x30)

[dmAuthdata]# This register serves as a 32-bit serial port to/from the authentication module.

When is clear, the debugger can communicate with the authentication module by reading or writing this register. There is no separate mechanism to signal overflow/underflow.

31	0
latexmath:[$	data
$]	
32	

12.14.17. Debug Module Control and Status 2 (`dmcs2`, at 0x32)

[dmDmcsTwo]# This register contains DM control and status bits that didn't easily fit in and . All are optional.

If halt groups are not implemented, then will always be 0 when is 0.

If resume groups are not implemented, then will remain 0 even after 1 is written there.

The DM external triggers available to add to halt groups may be the same as or distinct from the DM external triggers available to add to resume groups.

31	12	11		10	7	6	2	1		0	
latexmath:[$	0	$]		latexmath:[$	grouptype	$]		latexmath:[$	dmexttrigger	$]	
latexmath:[$	group	$]		latexmath:[$	hgwrite	$]		latexmath:[$	hgselect	$]	
20		1		4		5		1		1	

Field	Description	Access	Reset
Continued on next page			
[dmDmcsTwoGrouptype]# |grouptype|	0 (halt): The remaining fields in this register configure halt groups. 1 (resume): The remaining fields in this register configure resume groups.	WARL	0
[dmDmcsTwoDmexttrigger]# |dmexttrigger|	This field contains the currently selected DM external trigger. If a non-existent trigger value is written here, the hardware will change it to a valid one or 0 if no DM external triggers exist.	WARL	0
[dmDmcsTwoGroup]# |group|	When is 0, contains the group of the hart specified by latexmath:[$	\hyperref[hartsel]{hartsel}	$]. When is 1, contains the group of the DM external trigger selected by . The value written to this field is ignored unless is also written 1. Group numbers are contiguous starting at 0, with the highest number being implementation-dependent, and possibly different between different group types. Debuggers should read back this field after writing to confirm they are using a hart group that is supported. If groups aren't implemented, then this entire field is 0.

Field	Description	Access	Reset
WARL	preset	[dmDmcsTwoHgwrite]# |hgwrite|	When 1 is written and is 0, for every selected hart the DM will change its group to the value written to , if the hardware supports that group for that hart. Implementations may also change the group of a minimal set of unselected harts in the same way, if that is necessary due to a hardware limitation. When 1 is written and is 1, the DM will change the group of the DM external trigger selected by to the value written to , if the hardware supports that group for that trigger. Writing 0 has no effect.
W1	-	|hgselect|	0 (harts): Operate on harts. 1 (triggers): Operate on DM external triggers. If there are no DM external triggers, this field must be tied to 0.

12.14.18. Halt Summary 0 (`haltsum0`, at 0x40)

[dmHaltsumZero]# Each bit in this read-only register indicates whether one specific hart is halted or not. Unavailable/nonexistent harts are not considered to be halted.

This register might not be present if fewer than 2 harts are connected to this DM.

The LSB reflects the halt status of hart \{hartsel[19:5],5'h0}, and the MSB reflects halt status of hart \{hartsel[19:5],5'h1f}.

This entire register is read-only.

31	0
latexmath:[$	haltsum0
$]	
32	

12.14.19. Halt Summary 1 (`haltsum1`, at 0x13)

[dmHaltsumOne]# Each bit in this read-only register indicates whether any of a group of harts is halted or not. Unavailable/nonexistent harts are not considered to be halted.

This register might not be present if fewer than 33 harts are connected to this DM.

The LSB reflects the halt status of harts \{hartsel[19:10],10'h0} through \{hartsel[19:10],10'h1f}. The MSB reflects the halt status of harts \{hartsel[19:10],10'h3e0} through \{hartsel[19:10],10'h3ff}.

This entire register is read-only.

31	0
latexmath:[$	haltsum1
$]	
32	

12.14.20. Halt Summary 2 (`haltsum2`, at 0x34)

[dmHaltsumTwo]# Each bit in this read-only register indicates whether any of a group of harts is halted or not. Unavailable/nonexistent harts are not considered to be halted.

This register might not be present if fewer than 1025 harts are connected to this DM.

The LSB reflects the halt status of harts \{hartsel[19:15],15'h0} through \{hartsel[19:15],15'h3ff}. The MSB reflects the halt status of harts \{hartsel[19:15],15'h7c00} through \{hartsel[19:15],15'h7fff}.

This entire register is read-only.

31	0
latexmath:[$	haltsum2
$]	
32	

12.14.21. Halt Summary 3 (`haltsum3`, at 0x35)

[dmHaltsumThree]# Each bit in this read-only register indicates whether any of a group of harts is halted or not. Unavailable/nonexistent harts are not considered to be halted.

This register might not be present if fewer than 32769 harts are connected to this DM.

The LSB reflects the halt status of harts 20'h0 through 20'h7fff. The MSB reflects the halt status of harts 20'hf8000 through 20'hfffff.

This entire register is read-only.

31	0
latexmath:[$	haltsum3
$]	
32	

12.14.22. System Bus Access Control and Status (`sbcs`, at 0x38)

[dmSbcs]#

31	29	28	23	22		21		20	
latexmath :[$	sbversion	$]		latexmath :[$	0	$]		latexmath :[$	sbbusyerr or
$]		latexmath :[$	sbbusy	$]		latexmath :[$	sbreadona ddr	$]	
3		6		1		1		1	

19	17	16		15		14	12	11	5
latexmath :[$	sbaccess	$]		latexmath :[$	sbautoinc rement	$]		latexmath :[$	sbreadond ata
$]		latexmath :[$	sberror	$]		latexmath :[$	sbasize	$]	
3		1		1		3		7	

4		3		2		1		0	
latexmath :[$	sbaccess12 8	$]		latexmath :[$	sbaccess6 4	$]		latexmath :[$	sbaccess3 2
$]		latexmath :[$	sbaccess1 6	$]		latexmath :[$	sbaccess8	$]	
1		1		1		1		1	

Field	Description	Access	Reset
Continued on next page			
[dmSbcsSbversion]# |sbversion|	0 (legacy): The System Bus interface conforms to mainline drafts of this spec older than 1 January, 2018. 1 (1.0): The System Bus interface conforms to this version of the spec. Other values are reserved for future versions.	R	1
[dmSbcsSbbusyerror]# |sbbusyerror|	Set when the debugger attempts to read data while a read is in progress, or when the debugger initiates a new access while one is already in progress (while is set). It remains set until it's explicitly cleared by the debugger. While this field is set, no more system bus accesses can be initiated by the Debug Module.	R/W1C	0

Field	Description	Access	Reset
[dmSbcsSbbusy]# |sbbusy|	When 1, indicates the system bus manager is busy. (Whether the system bus itself is busy is related, but not the same thing.) This bit goes high immediately when a read or write is requested for any reason, and does not go low until the access is fully completed. Writes to while is high result in undefined behavior. A debugger must not write to until it reads as 0.	R	0
[dmSbcsSbreadonaddr]# |sbreadonaddr|	When 1, every write to automatically triggers a system bus read at the new address.	R/W	0
[dmSbcsSbaccess]# |sbaccess|	Select the access size to use for system bus accesses. 0 (8bit): 8-bit 1 (16bit): 16-bit 2 (32bit): 32-bit 3 (64bit): 64-bit 4 (128bit): 128-bit If has an unsupported value when the DM starts a bus access, the access is not performed and is set to 4.	R/W	2
[dmSbcsSbautoincrement]# |sbautoincrement|	When 1, `sbaddress` is incremented by the access size (in bytes) selected in after every system bus access.	R/W	0
[dmSbcsSbreadondata]# |sbreadondata|	When 1, every read from automatically triggers a system bus read at the (possibly auto-incremented) address.	R/W	0

Field	Description	Access	Reset
[dmSbcsSberror]# |sberror|	When the Debug Module's system bus manager encounters an error, this field gets set. The bits in this field remain set until they are cleared by writing 1 to them. While this field is non-zero, no more system bus accesses can be initiated by the Debug Module. An implementation may report `Other" (7) for any error condition. 0 (none): There was no bus error. 1 (timeout): There was a timeout. 2 (address): A bad address was accessed. 3 (alignment): There was an alignment error. 4 (size): An access of unsupported size was requested. 7 (other): Other.	R/W1C	0
[dmSbcsSbasize]# |sbasize|	Width of system bus addresses in bits. (0 indicates there is no bus access support.)	R	Preset
[dmSbcsSbaccessOneTwentyeight]# |sbaccess128|	1 when 128-bit system bus accesses are supported.	R	Preset
[dmSbcsSbaccessSixtyfour]# |sbaccess64|	1 when 64-bit system bus accesses are supported.	R	Preset
[dmSbcsSbaccessThirtytwo]# |sbaccess32|	1 when 32-bit system bus accesses are supported.	R	Preset
[dmSbcsSbaccessSixteen]# |sbaccess16|	1 when 16-bit system bus accesses are supported.	R	Preset
|sbaccess8|	1 when 8-bit system bus accesses are supported.	R	Preset

12.14.23. System Bus Address 31:0 (`sbaddress0`, at 0x39)

[dmSbaddressZero]# If is 0, then this register is not present.

When the system bus manager is busy, writes to this register will set and don't do anything else.

If is 0, is 0, and is set then writes to this register start the following:

1. Set .
2. Perform a bus read from the new value of `sbaddress`.
3. If the read succeeded and is set, increment `sbaddress`.
4. Clear .

31	0
latexmath:[$	address
$]	
32	

Field	Description	Access	Reset
Continued on next page			
|address|	Accesses bits 31:0 of the physical address in `sbaddress`.	R/W	0

12.14.24. System Bus Address 63:32 (`sbaddress1`, at 0x3a)

[dmSbaddressOne]# If is less than 33, then this register is not present.

When the system bus manager is busy, writes to this register will set and don't do anything else.

31	0
latexmath:[$	address
$]	
32	

Field	Description	Access	Reset
Continued on next page			
|address|	Accesses bits 63:32 of the physical address in `sbaddress` (if the system address bus is that wide).	R/W	0

12.14.25. System Bus Address 95:64 (`sbaddress2`, at 0x3b)

[dmSbaddressTwo]# If is less than 65, then this register is not present.

When the system bus manager is busy, writes to this register will set and don't do anything else.

31	0
latexmath:[$	address
$]	
32	

Field	Description	Access	Reset
Continued on next page			
|address|	Accesses bits 95:64 of the physical address in `sbaddress` (if the system address bus is that wide).	R/W	0

12.14.26. System Bus Address 127:96 (`sbaddress3`, at 0x37)

[dmSbaddressThree]# If is less than 97, then this register is not present.

When the system bus manager is busy, writes to this register will set and don't do anything else.

31	0
latexmath:[$	address
$]	
32	

Field	Description	Access	Reset
Continued on next page			
|address|	Accesses bits 127:96 of the physical address in `sbaddress` (if the system address bus is that wide).	R/W	0

12.14.27. System Bus Data 31:0 (`sbdata0`, at 0x3c)

[dmSbdataZero]# If all of the `sbaccess` bits in are 0, then this register is not present.

Any successful system bus read updates `sbdata`. If the width of the read access is less than the width of `sbdata`, the contents of the remaining high bits may take on any value.

If either or isn't 0 then accesses do nothing.

If the bus manager is busy then accesses set , and don't do anything else.

Writes to this register start the following:

1. Set .
2. Perform a bus write of the new value of `sbdata` to `sbaddress`.
3. If the write succeeded and is set, increment `sbaddress`.
4. Clear .

Reads from this register start the following:

1. ``Return" the data.
2. Set .
3. If is set:

a. Perform a system bus read from the address contained in `sbaddress`, placing the result in `sbdata`.

b. If is set and the read was successful, increment `sbaddress`.

4. Clear .

Only has this behavior. The other `sbdata` registers have no side effects. On systems that have buses wider than 32 bits, a debugger should access after accessing the other `sbdata` registers.

31	0
latexmath:[$	data
$]	
32	

Field	Description	Access	Reset
Continued on next page			
|data|	Accesses bits 31:0 of `sbdata`.	R/W	0

12.14.28. System Bus Data 63:32 (`sbdata1`, at 0x3d)

[dmSbdataOne]# If and are 0, then this register is not present.

If the bus manager is busy then accesses set , and don't do anything else.

31	0
latexmath:[$	data
$]	
32	

Field	Description	Access	Reset
Continued on next page			
|data|	Accesses bits 63:32 of `sbdata` (if the system bus is that wide).	R/W	0

12.14.29. System Bus Data 95:64 (`sbdata2`, at 0x3e)

[dmSbdataTwo]# This register only exists if is 1.

If the bus manager is busy then accesses set , and don't do anything else.

31	0
latexmath:[$	data
$]	
32	

Field	Description	Access	Reset
Continued on next page			
|data|	Accesses bits 95:64 of `sbdata` (if the system bus is that wide).	R/W	0

12.14.30. System Bus Data 127:96 (`sbdata3`, at 0x3f)

[dmSbdataThree]# This register only exists if is 1.

If the bus manager is busy then accesses set , and don't do anything else.

31	0
latexmath:[$	data
$]	
32	

Field	Description	Access	Reset
Continued on next page			
|data|	Accesses bits 127:96 of `sbdata` (if the system bus is that wide).	R/W	0

12.14.31. Custom Features (`custom`, at 0x1f)

[dmCustom]# This optional register may be used for non-standard features. Future version of the debug spec will not use this address.

12.14.32. Custom Features 0 (`custom0`, at 0x70)

[dmCustomZero]# The optional through registers may be used for non-standard features. Future versions of the debug spec will not use these addresses.

Chapter 13. Sdext (ISA Extension)

This chapter describes the Sdext ISA extension. It must be implemented to make external debug work, and is only useful in conjunction with external debug.

Modifications to the RISC-V core to support debug are kept to a minimum. There is a special execution mode (Debug Mode) and a few extra CSRs. The DM takes care of the rest.

In order to be compatible with this specification an implementation must implement everything described in this chapter that is not explicitly listed as optional.

If Sdext is implemented and Sdtrig is not implemented, then accessing any of the Sdtrig CSRs must raise an illegal instruction exception.

13.1. Debug Mode

Debug Mode is a special processor mode used only when a hart is halted for external debugging. Because the hart is halted, there is no forward progress in the normal instruction stream. How Debug Mode is implemented is not specified here.

When executing code due to an abstract command, the hart stays in Debug Mode and the following apply:

1. All implemented instructions operate just as they do in M-mode, unless an exception is mentioned in this list.
2. All operations are executed with machine mode privilege, except that additional Debug Mode CSRs are accessible and $|MPRV|$ in `mstatus` may be ignored according to . Full permission checks, or a relaxed set of permission checks, will apply according to .
3. All interrupts (including NMI) are masked.
4. Traps don't take place. Instead, they end execution of the program buffer and the hart remains in Debug Mode. Because they do not trap to M-mode, they do not update registers such as `mstatus`, `mepc`, `mcause`, `` mtval`, `mtval2`, and `mtinst`. The same is true for the equivalent privileged registers that are updated when trapping to other modes. Registers that may be updated as part of execution before the exception are allowed to be updated. For example, vector load/store instructions which raise exceptions may partially update the destination register and set `vstart` appropriately.
5. Triggers don't match or fire.
6. If is 0 then counters continue. If it is 1 then counters are stopped.
7. If is 0 then `time` continues to update. If it is 1 then `time` will not update. It will resynchronize with `mtime` after leaving Debug Mode.
8. Instructions that place the hart into a stalled state act as a `nop`. This includes `wfi`, `wrs.sto`, and `wrs.nto`.
9. Almost all instructions that change the privilege mode have unspecified behavior. This includes `ecall`, `mret`, `sret`, and `uret`. (To change the privilege mode, the debugger can write and in). The only exception is `ebreak`, which ends execution of the Program Buffer when executed.
10. All control transfer instructions may act as illegal instructions if their destination is in the Program Buffer. If one such instruction acts as an illegal instruction, all such instructions must act as illegal instructions.

11. All control transfer instructions may act as illegal instructions if their destination is outside the Program Buffer. If one such instruction acts as an illegal instruction, all such instructions must act as illegal instructions.
12. Instructions that depend on the value of the PC (e.g. `auipc`) may act as illegal instructions.
13. Effective XLEN is DXLEN.
14. Forward progress is guaranteed.

> — —
>
> ```
>  +
> When , the external debugger can set MPRV and MPP appropriately to have
> hardware perform memory accesses with the appropriate endianness,
> address translation, permission checks, and PMP/PMA checks (subject to
> ). This is also the only way to access all of physical memory when
> 34-bit physical addresses are supported on a Sv32 hart. If hardware ties
> to 0 then the external debugger is expected to simulate all the effects
> of MPRV, including any extensions that affect memory accesses. For these
> reasons it is recommended to tie to 1.
> ```

13.2. Load-Reserved/Store-Conditional Instructions

The reservation registered by an `lr` instruction on a memory address may be lost when entering Debug Mode or while in Debug Mode. This means that there may be no forward progress if Debug Mode is entered between `lr` and `sc` pairs.

> — —
>
> ```
>  +
> This is a behavior that debug users must be aware of. If they have a
> breakpoint set between a `lr` and `sc` pair, or are stepping through
> such code, the `sc` may never succeed. Fortunately in general use there
> will be very few instructions in such a sequence, and anybody debugging
> it will quickly notice that the reservation is not occurring. The
> solution in that case is to set a breakpoint on the first instruction
> after the `sc` and run to it. A higher level debugger may choose to
> automate this.
> ```

13.3. Wait for Interrupt Instruction

If halt is requested while `wfi` is executing, then the hart must leave the stalled state, completing this instruction's execution, and then enter Debug Mode.

13.4. Wait-on-Reservation-Set Instructions

If halt is requested while `wrs.sto` or `wrs.nto` is executing, then the hart must leave the stalled state, completing this instruction's execution, and then enter Debug Mode.

13.5. Single Step

13.5.1. Step Bit In Dcsr

This method is only available to external debuggers, and is the preferred way to single step.

An external debugger can cause a halted hart to execute a single instruction or trap and then re-enter Debug Mode by setting before resuming. If is set when a hart resumes then it will single step, regardless of the reason for resuming.

If control is transferred to a trap handler while executing the instruction, then Debug Mode is re-entered immediately after the PC is changed to the trap handler, and the appropriate `tval` and `cause` registers are updated. In this case none of the trap handler is executed, and if the cause was a pending interrupt no instructions might be executed at all.

If executing or fetching the instruction causes a trigger to fire with action=1, Debug Mode is re-entered immediately after that trigger has fired. In that case is set to 2 (trigger) instead of 4 (single step). Whether the instruction is executed or not depends on the specific configuration of the trigger.

If the instruction that is executed causes the PC to change to an address where an instruction fetch causes an exception, that exception does not occur until the next time the hart is resumed. Similarly, a trigger at the new address does not fire until the hart actually attempts to execute that instruction.

If the instruction being stepped over would normally stall the hart, then instead the instruction is treated as a `nop`. This includes `wfi`, `wrs.sto`, and `wrs.nto`.

13.5.2. Icount Trigger

Native debuggers won't have access to , but can use the trigger by setting to 1.

This approach does have some limitations:

1. Interrupts will fire as usual. Debuggers that want to disable interrupts while stepping must disable them by changing `mstatus`, and specially handle instructions that read `mstatus`.
2. `wfi` instructions are not treated specially and might take a very long time to complete.

This mechanism cleanly supports a system which supports multiple privilege levels, where the OS or a debug stub runs in M-Mode while the program being debugged runs in a less privileged mode. Systems that only support M-Mode can use as well, but count must be able to count several instructions (depending on the software implementation). See Section 9.3.1.

13.6. Reset

If the halt signal (driven by the hart's halt request bit in the Debug Module) or |hyperref [*resethaltreq*{resethaltreq}|$] are asserted when a hart comes out of reset, the hart must enter Debug Mode before executing any instructions, but after performing any initialization that would usually happen before the first instruction is executed.

13.7. Halt

When a hart halts:

1. is updated.
2. and are set to reflect current privilege mode.
3. is set to the next instruction that should be executed.
4. If the current instruction can be partially executed and should be restarted to complete, then the relevant state for that is updated. E.g. if a halt occurs during a partially executed vector instruction, then `vstart` is updated, and is updated to the address of the partially executed instruction. This is analogous to how vector instructions behave for exceptions.
5. The hart enters Debug Mode.

13.8. Resume

When a hart resumes:

1. `pc` changes to the value stored in .
2. The current privilege mode and virtualization mode are changed to that specified by and .
3. If the new privilege mode is less privileged than M-mode, $|MPRV|$ in `mstatus` is cleared.
4. The hart is no longer in debug mode.

13.9. XLEN

While in Debug Mode, XLEN is DXLEN. It is up to the debugger to determine the XLEN during normal program execution (by looking at `misa`) and to clearly communicate this to the user.

13.10. Core Debug Registers

The supported Core Debug Registers must be implemented for each hart that can be debugged. They are CSRs, accessible using the RISC-V `csr` opcodes and optionally also using abstract debug commands.

Attempts to access an unimplemented Core Debug Register raise an illegal instruction exception.

13.11. Virtual Debug Registers

Chapter 14. Sdtrig (ISA Extension)

This chapter describes the Sdtrig ISA extension, which can be implemented independently of functionality described in the other chapters. It consists exclusively of the Trigger Module ™.

Triggers can cause a breakpoint exception, entry into Debug Mode, or a trace action without having to execute a special instruction. This makes them invaluable when debugging code from ROM. They can trigger on execution of instructions at a given memory address, or on the address/data in loads/stores.

If Sdtrig is implemented, the Trigger Module must support at least one trigger. Accessing trigger CSRs that are not used by any of the implemented triggers must result in an illegal instruction exception. M-Mode and Debug Mode accesses to trigger CSRs that are used by any of the implemented triggers must succeed, regardless of the current type of the currently selected trigger.

A trigger matches when the conditions that it specifies (e.g. a load from a specific address) are met. A trigger fires when a trigger that matches performs the action configured for that trigger.

Triggers do not fire while in Debug Mode.

14.1. Enumeration

Each trigger may support a variety of features. A debugger can build a list of all triggers and their features as follows:

1. Write 0 to . If this results in an illegal instruction exception, then there are no triggers implemented.
2. Read back and check that it contains the written value. If not, exit the loop.
3. Read .
4. If that caused an exception, the debugger must read to discover the type. (If is 0, this trigger doesn't exist. Exit the loop.)
5. If is 1, this trigger doesn't exist. Exit the loop.
6. Otherwise, the selected trigger supports the types discovered in .
7. Repeat, incrementing the value in .

> – –
>
> ---
>
> ```
>  +
> The above algorithm reads back so that implementations which have
> latexmath:[$2^n$] triggers only need to implement latexmath:[$n$] bits
> of .
> ```
>
> The algorithm checks and in case the implementation has m bits of but fewer than 2^m triggers.

14.2. Actions

Triggers can be configured to take one of several actions when they fire. Table #tab:action lists all options.

|r|L| Value & Description
& Raise a breakpoint exception. (Used when software wants to use the trigger module without an external debugger attached.) `xepc` must contain the virtual address of the next instruction that must be executed to preserve the program flow.
& Enter Debug Mode. must contain the virtual address of the next instruction that must be executed to preserve the program flow.

This action is only legal when the trigger's is 1. Since the `tdata` registers are WARL, hardware should clear the action field whenever the action field is 1, the new value of would be 0, and the new value of the action field would be 1.

This action can only be supported if Sdext is implemented on the hart.
& Trace on, described in the trace specification.
& Trace off, described in the trace specification.
& Trace notify, described in the trace specification.
& Reserved for use by the trace specification.
– 9 & Send a signal to TM external trigger output 0 or 1 (respectively).
other & Reserved for future use.

> – –
>
> ```
>  +
> Actions 8 and 9 are intended to increment custom event counters, but
> these signals could also be brought to outputs for use by external
> logic.
> ```

14.3. Priority

Table #tab:priority lists the synchronous exceptions from the Privileged Spec, and where the various types of triggers fit in. The first 3 columns come from the Privileged Spec, and the final column shows where triggers fit in. Priorities in the table are separated by horizontal lines, so e.g. etrigger and itrigger have the same priority. If this table contradicts the table in the Privileged Spec, then the latter takes precedence.

This table only applies if triggers are precise. Otherwise triggers will fire some indeterminate time after the event, and the priority is irrelevant. When triggers are chained, the priority is the lowest priority of the triggers in the chain.

|l|p.7in|p2.3in|p2.5in| Priority & Exception Code & Description & Trigger
Highest & 3 & & etrigger
& 3 & & icount
& 3 & & itrigger
& 3 & & mcontrol/mcontrol6 after (on previous instruction)
& 3 & Instruction address breakpoint & mcontrol/mcontrol6 execute address before

& 12, 20, 1 & During instruction address translation: First encountered page fault, guest-page fault, or access fault &
& 1 & With physical address for instruction: Instruction access fault &
& 3 & & mcontrol/mcontrol6 execute data before
& 2 & Illegal instruction &
& 22 & Virtual instruction &
& 0 & Instruction address misaligned &
& 8, 9, 10, 11 & Environment call &
& 3 & Environment break &
& 3 & Load/Store/AMO address breakpoint & mcontrol/mcontrol6 load/store address/data before
& 4, 6 & Optionally: Load/Store/AMO address misaligned &
& 13, 15, 21, 23, 5, 7 & During address translation for an explicit memory access: First encountered page fault, guest-page fault, or access fault &
& 5, 7 & With physical address for an explicit memory access: Load/store/AMO access fault &
& 4, 6 & If not higher priority: Load/store/AMO address misaligned &
Lowest & 3 & & mcontrol/mcontrol6 load data before

When multiple triggers in the same priority fire at once, (if implemented) is set for all of them. If more than one of these triggers has then `tval` is updated in accordance with one of them, but which one is unspecified. If one of these triggers has the `enter Debug Mode'' action (1) and another trigger has the` raise a breakpoint exception" action (0), the preferred behavior is to have both actions take place. It is implementation-dependent which of the two happens first. This ensures both that the presence of an external debugger doesn't affect execution and that a trigger set by user code doesn't affect the external debugger. If this is not implemented, then the hart must enter Debug Mode and ignore the breakpoint exception. In the latter case, of the trigger whose action is 0 must still be set, giving a debugger an opportunity to handle this case. What happens with trace actions when triggers with different actions are also firing is left to the trace specification.

14.4. Native Triggers

Triggers can be used for native debugging when . If supported by the hart and desired by the debugger, triggers will often be programmed to have so that when they fire they cause a breakpoint exception to trap to a more privileged mode. That breakpoint exception can either be taken in M-mode or it can be delegated to a less privileged mode. However, it is possible for triggers to fire in the same mode that the resulting exception will be handled in.

In these cases such a trigger may cause a breakpoint exception while already in a trap handler. This might leave the hart unable to resume normal execution because state such as `mcause` and `mepc` would be overwritten.

> – –
>
> ```
>  +
> In particular, when :
> ```
>
> 1. mcontrol and mcontrol6 triggers with can cause a breakpoint exception that is taken from M-mode to M-mode (regardless of delegation).
> 2. mcontrol and mcontrol6 triggers with can cause a breakpoint exception that is

taken from S-mode to S-mode if `medeleg`[3]=1.

3. mcontrol6 triggers with can cause a breakpoint exception that is taken from VS-mode to VS-mode if `medeleg`[3]=1 and `hedeleg`[3]=1.
4. icount triggers with can cause a breakpoint exception that is taken from M-mode to M-mode (regardless of delegation).
5. icount triggers with can cause a breakpoint exception that is taken from S-mode to S-mode if `medeleg`[3]=1.
6. icount triggers with can cause a breakpoint exception that is taken from VS-mode to VS-mode if `medeleg`[3]=1 and `hedeleg`[3]=1.
7. etrigger and itrigger triggers will always be taken from a trap handler before the first instruction of the handler. If etrigger/itrigger is set to trigger on exception/interrupt X and if X is delegated to mode Y then the trigger will cause a breakpoint exception that is taken from mode Y to mode Y unless breakpoint exceptions are delegated to a more privileged mode than Y.
8. tmexttrigger triggers are asynchronous and may occur in any mode and at any time.

Harts that support triggers with should implement one of the following two solutions to solve the problem of reentrancy:

1. The hardware prevents triggers with from matching or firing while in M-mode and while $|MIE|$ in `mstatus` is 0. If `medeleg`[3]=1 then it prevents triggers with from matching or firing while in S-mode and while $|SIE|$ in `sstatus` is 0. If `medeleg`[3]=1 and `hedeleg`[3]=1 then it prevents triggers with from matching or firing while in VS-mode and while $|SIE|$ in `vsstatus` is 0.
2. and in is implemented. `medeleg`[3] is hard-wired to 0.

– –

```
 +
The first option has the limitation that interrupts might be disabled at
times when a user still might want triggers to fire. It has the benefit
that breakpoints are not required to be handled in M-mode.
```

The second option has the benefit that it only disables triggers during the trap handler, though it requires specific software support for this debug feature in the M-mode trap handlers. It can only work if breakpoints are not delegated to less privileged modes and therefore targets primarily implementations without S-mode.

Because is not accessible to S-mode, the second option can not be extended to accommodate delegation without adding additional S-mode and VS-mode CSRs.

Both options prevent etrigger and itrigger from having any effect on exceptions and interrupts that are handled in M-mode. They also prevent triggering during some initial portion of each handler. Debuggers should use other mechanisms to debug

> these cases, such as patching the handler or setting a breakpoint on the instruction after $|MIE|$ is cleared.

14.5. Memory Access Triggers

and both enable triggers on memory accesses. This section describes for both of them how certain corner cases are treated.

14.5.1. A Extension

If the A extension is supported, then triggers on loads/stores treat them as follows:

1. `lr` instructions are loads.
2. Successful `sc` instructions are stores.
3. It is unspecified whether failing `sc` instructions are stores or not.
4. Each AMO instruction is a load for the read portion of the operation. The address is always available to trigger on, although the value loaded might not be, depending on the hardware implementation.
5. Each AMO instruction is a store for the write portion of the operation. The address is always available to trigger on, although the value stored might not be, depending on the hardware implementation.

If the destination register of any load or AMO is `zero` then it is unspecified whether a data load trigger will match. Whether data store triggers match on AMOs is unspecified.

14.5.2. Combined Accesses

Some instructions lead a hart to perform multiple memory accesses. This includes vector loads and stores, as well as `cm.push` and `cm.pop` instructions. The Trigger Module should match such accesses as if they all happened individually. E.g. a vector load should be treated as if it performed multiple loads of size SEW (selected element width), and `cm.push` should be treated as if it performed multiple stores of size XLEN.

14.5.3. Cache Operations

Cache operations are infrequently performed, and code that uses them can have hard-to-find bugs. For the purposes of debug triggers, two classes of cache operations must match as stores:

1. Cache operations that enable software to maintain coherence between otherwise non-coherent implicit and explicit memory accesses.
2. Cache operations that perform block writes of constant data.

Only triggers with and will match. Since cache operations affect multiple addresses, there are multiple possible values to compare against. Implementations must implement one of the following options. From most desirable to least desirable, they are:

1. Every address from the effective address rounded down to the nearest cache block boundary (inclusive) to the effective address rounded up to the nearest cache block boundary (exclusive) is a compare value.

2. The effective address rounded down to the nearest cache block boundary is a compare value.
3. The effective address of the instruction is a compare value.

Cache operations encoded as HINTs do not match debug triggers.

> — —
>
> ```
>  +
> The above language intends to capture the trigger behavior with respect
> to the cache operations to be introduced in a forthcoming I/D
> consistency extension.
> ```
>
> For RISC-V Base Cache Management Operation ISA Extensions 1.0.1, this means the following:
>
> 1. `cbo.clean`, `cbo.flush`, and `cbo.inval` match as if they are stores because they affect consistency.
> 2. `cbo.zero` matches as if it is a store because it performs a block write of constant data.
> 3. The prefetch instructions don't match at all.

14.6. Multiple State Change Instructions

An instruction that performs multiple architectural state changes (e.g., register updates and/or memory accesses) might cause a trigger to fire at an intermediate point in its execution. As a result, architectural state changes up to that point might have been performed, while subsequent state changes, starting from the event that activated the trigger, might not have been. The definition of such an instruction will specify the order in which architectural state changes take place. Alternatively, it may state that partial execution is not allowed, implying that a mid-execution trigger must prevent any architectural state changes from occurring.

Debuggers won't be aware if an instruction has been partially executed. When they resume execution, they will execute the same instruction once more. Therefore, it's crucial that partially executing the instruction and then executing it again leaves the hart in a state closely resembling the state it would have been in if the instruction had only been executed once.

14.7. Trigger Registers

These registers are CSRs, accessible using the RISC-V `csr` opcodes and optionally also using abstract debug commands.

Almost all trigger functionality is optional. All `tdata` registers follow write-any-read-legal semantics. If a debugger writes an unsupported configuration, the register will read back a value that is supported (which may simply be a disabled trigger). This means that a debugger must always read back values it writes to `tdata` registers, unless it already knows already what is supported. Writes to one `tdata` register must not modify the contents of other `tdata` registers, nor the configuration of any trigger besides the one that is currently selected.

The combination of these rules means that a debugger cannot simply set a trigger by writing , then , etc. The current value of might not be legal with the new value of . To help with this situation, it is guaranteed that writing 0 to disables the trigger, and leaves it in a state where and can be written with any value that makes sense for any trigger type supported by this trigger.

As a result, a debugger can write any supported trigger as follows:

1. Write 0 to . (This will result in containing a non-zero value, since the register is **WARL**.)
2. Write desired values to and .
3. Write desired value to .

Code that restores CSR context of triggers that might be configured to fire in the current privilege mode must use this same sequence to restore the triggers. This avoids the problem of a partially written trigger firing at a different time than is expected.

Attempts to access an unimplemented Trigger Register raise an illegal instruction exception.

Chapter 15. Debug Transport Module (DTM) (non-ISA extension)

Debug Transport Modules provide access to the DM over one or more transports (e.g. JTAG or USB).

There may be multiple DTMs in a single hardware platform. Ideally every component that communicates with the outside world includes a DTM, allowing a hardware platform to be debugged through every transport it supports. For instance a USB component could include a DTM. This would trivially allow any hardware platform to be debugged over USB. All that is required is that the USB module already in use also has access to the Debug Module Interface.

Using multiple DTMs at the same time is not supported. It is left to the user to ensure this does not happen.

This specification defines a JTAG DTM in Section #sec:jtagdtm[7.1]. Additional DTMs may be added in future versions of this specification.

An implementation can be compatible with this specification without implementing any of this section. In that case it must be advertised as conforming to `RISC-V Debug Specification 1.0-, with custom DTM.'' If the JTAG DTM described here is implemented, it must be advertised as conforming to the` RISC-V Debug Specification 1.0-, with JTAG DTM."

15.1. JTAG Debug Transport Module

This Debug Transport Module is based around a normal JTAG Test Access Port (TAP). The JTAG TAP allows access to arbitrary JTAG registers by first selecting one using the JTAG instruction register (IR), and then accessing it through the JTAG data register (DR).

15.1.1. JTAG Background

JTAG refers to IEEE Std 1149.1-2013. It is a standard that defines test logic that can be included in an integrated circuit to test the interconnections between integrated circuits, test the integrated circuit itself, and observe or modify circuit activity during the component's normal operation. This specification uses the latter functionality. The JTAG standard defines a Test Access Port (TAP) that can be used to read and write a few custom registers, which can be used to communicate with debug hardware in a component.

15.1.2. JTAG DTM Registers

JTAG TAPs used as a DTM must have an IR of at least 5 bits. When the TAP is reset, IR must default to 00001, selecting the IDCODE instruction. A full list of JTAG registers along with their encoding is in Table #dtmTable:jtagregisters. If the IR actually has more than 5 bits, then the encodings in Table #dtmTable:jtagregisters should be extended with 0's in their most significant bits, except for the 0x1f encoding of BYPASS, which must be extended with 1's in the most significant bits. The only regular JTAG registers a debugger might use are BYPASS and IDCODE, but this specification leaves IR space for many other standard JTAG instructions. Unimplemented instructions must select the BYPASS register.

15.1.3. JTAG Connector

15.1.3.1. Recommended JTAG Connector

To make it easy to acquire debug hardware, this spec recommends a connector that is compatible with the MIPI-10 .05 inch connector specification, as described in MIPI Debug & Trace Connector Recommendations, Version 1.20, 2 July 2021.

The connector has .05 inch spacing, gold-plated male header with .016 inch thick hardened copper or beryllium bronze square posts (SAMTEC FTSH or equivalent). Female connectors are compatible $20\mu m$ gold connectors.

Viewing the male header from above (the pins pointing at your eye), a target's connector looks as it does in Table #tab:mipiten[7.1]. The function of each pin is described in Table #tab:pinout.

VREF DEBUG	1	2	TMS
GND	3	4	TCK
GND	5	6	TDO
GND or KEY	7	8	TDI
GND	9	10	nRESET

Table 8. MIPI 10-pin JTAG + nRESET Connector Diagram

|l|c|L| Essential & GND & Connected to ground.
& TCK & JTAG TCK signal, driven by the debug adapter.
& TDI & JTAG TDI signal, driven by the debug adapter.
& TDO & JTAG TDO signal, driven by the target.
& TMS & JTAG TMS signal, driven by the debug adapter.
& VREF DEBUG & Reference voltage for logic high.
Recommended & nRESET & Open drain active low reset signal, usually driven by the debug adapter. The signal may be used bi-directional to drive or sense the target reset signal.

Asserting reset should reset any RISC-V cores as well as any other peripherals on the PCB. It should not reset the debug logic. This pin is optional but strongly encouraged.

nRESET should never be connected to the TAP reset, otherwise the debugger might not be able to debug through a reset to discover the cause of a crash or to maintain execution control after the reset.
& KEY & This pin may be cut on the male and plugged on the female header to ensure the header is always plugged in correctly. It is, however, recommended to use this pin as an additional ground, to allow for fastest TCK speeds. A shrouded connector should be used to prevent the cable from being plugged in incorrectly.
Advanced & EXT & Reserved for custom use. Could be an input or an output.
& TRIGIN & Not used by this specification, to be driven by debug adapter. (Can be used for extended functions like UART or boot mode selection by some debug adapters).
& TRIGOUT & Not used by this specification, driven by the target.
Specialized & nTRST & Test reset, driven by the debug adapter. Asserting nTRST initializes the JTAG DTM asynchronously. It is used in systems where the JTAG DTM is not ready to be used after a normal power up. This signal is sometimes called TRST*.
Legacy & RTCK & Return test clock, driven by the target. A target may relay the TCK signal here once it has processed it, allowing a debugger to adjust its TCK frequency in response.

This signal should only be used to support legacy components that rely on this functionality.

& nTRST_PD & Test reset pull-down, driven by the debug adapter. Same function as nTRST, but with pull-down resistor on target.

This signal should only be used to support legacy components that rely on this functionality.

If a hardware platform requires nTRST then it is permissible to reuse the nRESET pin as the nTRST signal, resulting in a MIPI 10-pin JTAG
nTRST connector.

15.1.3.2. Alternate JTAG Connector

The MIPI-10 connector should provide plenty of signals for all modern hardware. If a design does need legacy JTAG signals, then the MIPI-20 connector should be used. Pins whose functionality isn't needed may be left unconnected.

Its physical connector is virtually identical to MIPI-10, except that it's twice as long, supporting twice as many pins. Its pinout is shown in Table #tab:mipitwenty[7.2]. The function of each pin is described in Table #tab:pinout.

VREF DEBUG	1	2	TMS
GND	3	4	TCK
GND	5	6	TDO
GND or KEY	7	8	TDI
GND	9	10	nRESET
GND	11	12	GND or RTCK
GND	13	14	NC or nTRST_PD
GND	15	16	nTRST or NC
GND	17	18	TRIGIN or NC
GND	19	20	TRIGOUT or GND

Table 9. MIPI 20-pin JTAG Connector Diagram

15.1.4. cJTAG

This spec does not have specific recommendations on how to use the cJTAG protocol.

When implementing cJTAG access to a JTAG DTM, the MIPI 10-pin Narrow JTAG connector should be used. Pins whose functionality isn't needed may be left unconnected.

Viewing the male header from above (the pins pointing at your eye), a target's connector looks as it does in Table #tab:mipicjtag[7.3].

VREF DEBUG	1	2	TMSC
GND	3	4	TCKC
GND	5	6	EXT or NC
GND or KEY	7	8	NC or nTRST_PD
GND	9	10	nRESET

Table 10. MIPI 10-pin Narrow JTAG Connector Diagram

Chapter 16. Hardware Implementations

Below are two possible implementations. A designer could choose one, mix and match, or come up with their own design.

16.1. Abstract Command Based

Halting happens by stalling the hart execution pipeline.

Muxes on the register file(s) allow for accessing GPRs and CSRs using the Access Register abstract command.

Memory is accessed using the Abstract Access Memory command or through System Bus Access.

This implementation could allow a debugger to collect information from the hart even when that hart is unable to execute instructions.

16.2. Execution Based

This implementation only implements the Access Register abstract command for GPRs on a halted hart, and relies on the Program Buffer for all other operations. It uses the hart's existing pipeline and ability to execute from arbitrary memory locations to avoid modifications to a hart's datapath.

When the halt request bit is set, the Debug Module raises a special interrupt to the selected harts. This interrupt causes each hart to enter Debug Mode and jump to a defined memory region that is serviced by the DM and is only accessible to the harts in Debug Mode. Accesses to this memory should be uncached to avoid side effects from debugging operations. When taking this jump, `pc` is saved to and is updated in . This jump is similar to a trap but it is not architecturally considered a trap, so for instance doesn't count as a trap for trigger behavior.

The code in the Debug Module causes the hart to execute a `` `park loop.'' In the park loop the hart writes its `mhartid `` to a memory location within the Debug Module to indicate that it is halted. To allow the DM to individually control one out of several halted harts, each hart polls for flags in a DM-controlled memory location to determine whether the debugger wants it to execute the Program Buffer or perform a resume.

To execute an abstract command, the DM first populates some internal words of program buffer according to . When is set, the DM populates these words with `lw <gpr>, 0x400(zero)` or `sw 0x400(zero), <gpr>`. 64- and 128-bit accesses use `ld`/`sd` and `lq`/`sq` respectively. If is not set, the DM populates these instructions as `nop`s. If is set, execution continues to the debugger-controlled Program Buffer, otherwise the DM causes a `ebreak` to execute immediately.

When `ebreak` is executed (indicating the end of the Program Buffer code) the hart returns to its park loop. If an exception is encountered, the hart jumps to an address within the Debug Module. The code there causes the hart to write to the Debug Module indicating an exception. Then the hart jumps back to the park loop. The DM infers from the write that there was an exception, and sets appropriately. Typically the hart will execute a `fence` instruction before entering the park loop, to ensure that any effects from the abstract command, such as a write to , take effect before the DM returns to 0.

To resume execution, the debug module sets a flag which causes the hart to execute a `dret`. `dret` is an instruction that only has meaning while in Debug Mode and not executing from the Program Buffer.

Its recommended encoding is 0x7b200073. When `dret` is executed, `pc` is restored from and normal execution resumes at the privilege set by and .

etc. are mapped into regular memory at an address relative to `zero` with only a 12-bit `imm`. The exact address is an implementation detail that a debugger must not rely on. For example, the `data` registers might be mapped to `0x400`.

For additional flexibility, , etc. are mapped into regular memory immediately preceding , in order to form a contiguous region of memory which can be used for either program execution or data transfer.

The PMP must not disallow fetches, loads, or stores in the address range associated with the Debug Module when the hart is in Debug Mode, regardless of how the PMP is configured. The same is true of PMA. Without this guarantee, the park loop would enter an infinite loop of traps and debug would not be possible.

16.3. Debug Module Interface Signals

As stated in section 4.1 the details of the DMI are left to the system designer. It is quite often the case that only one DTM and one DM is implemented. In this case it might be useful to comply with the signals suggested in table #tab:dmi_signals[8.1], which is the implementation used in the open-source rocket-chip RISC-V core.

The DTM can start a request when the DM sets REQ_READY to 1. When this is the case REQ_OP can be set to 1 for a read or 2 for a write request. The desired address is driven with the REQ_ADDRESS signal. Finally REQ_VALID is set high, indicating to the DM that a valid request is pending.

The DM must respond to a request from the DTM when RSP_READY is high. The status of the response is indicated by the RSP_OP signal (see). The data of the response is driven to RSP_DATA. A pending response is signalled by setting RSP_VALID.

Signal	Width	Source	Description
REQ_VALID	1	DTM	Indicates that a valid request is pending
REQ_READY	1	DM	Indicates that the DM is able to process a request
REQ_ADDRESS		DTM	Requested address
REQ_DATA	32	DTM	Requested data
REQ_OP	2	DTM	Same meaning as the field
RSP_VALID	1	DM	Indicates that a valid respond is pending
RSP_READY	1	DTM	Indicates that the DTM is able to process a respond
RSP_DATA	32	DM	Response data
RSP_OP	2	DM	Same meaning as the field

Table 11. Signals for the suggested DMI between one DTM and one DM

Chapter 17. Debugger Implementation

17.1. C Header File

github.com/riscv/riscv-debug-spec contains instructions for generating a C header file that defines macros for every field in every register/abstract command mentioned in this document.

17.2. External Debugger Implementation

This section details how an external debugger might use the described debug interface to perform some common operations on RISC-V cores using the JTAG DTM described in Section #sec:jtagdtm[7.1]. All these examples assume a 32-bit core but it should be easy to adapt the examples to 64- or 128-bit cores.

To keep the examples readable, they all assume that everything succeeds, and that they complete faster than the debugger can perform the next access. This will be the case in a typical JTAG setup. However, the debugger must always check the sticky error status bits after performing a sequence of actions. If it sees any that are set, then it should attempt the same actions again, possibly while adding in some delay, or explicit checks for status bits.

17.2.1. Debug Module Interface Access

To read an arbitrary Debug Module register, select , and scan in a value with set to 1, and set to the desired register address. In Update-DR the operation will start, and in Capture-DR its results will be captured into . If the operation didn't complete in time, will be 3 and the value in must be ignored. The busy condition must be cleared by writing in , and then the second scan scan must be performed again. This process must be repeated until returns 0. In later operations the debugger should allow for more time between Update-DR and Capture-DR.

To write an arbitrary Debug Bus register, select , and scan in a value with set to 2, and and set to the desired register address and data respectively. From then on everything happens exactly as with a read, except that a write is performed instead of the read.

It should almost never be necessary to scan IR, avoiding a big part of the inefficiency in typical JTAG use.

17.2.2. Checking for Halted Harts

A user will want to know as quickly as possible when a hart is halted (e.g. due to a breakpoint). To efficiently determine which harts are halted when there are many harts, the debugger uses the `haltsum` registers. Assuming the maximum number of harts exist, first it checks . For each bit set there, it writes |hyperref [*hartsel*{hartsel}|$], and checks . This process repeats through and . Depending on how many harts exist, the process should start at one of the lower `haltsum` registers.

17.2.3. Halting

To halt one or more harts, the debugger selects them, sets , and then waits for to indicate the harts are halted. Then it can clear to 0, or leave it high to catch a hart that resets while halted.

17.2.4. Running

First, the debugger should restore any registers that it has overwritten. Then it can let the selected harts run by setting . Once is set, the debugger knows the selected harts have resumed. Harts might halt very quickly after resuming (e.g. by hitting a software breakpoint) so the debugger cannot use /to check whether the hart resumed.

17.2.5. Single Step

Using the hardware single step feature is almost the same as regular running. The debugger just sets in before letting the hart run. The hart behaves exactly as in the running case, except that interrupts may be disabled (depending on) and it only fetches and executes a single instruction before re-entering Debug Mode.

17.2.6. Accessing Registers

17.2.6.1. Using Abstract Command

Read `s0` using abstract command:

|c|r|p0.3|p0.3| Op & Address & Value & Comment
Write & & = 2, , = 0x1008 & Read `s0`
Read & & - & Returns value that was in `s0`

Write `mstatus` using abstract command:

|c|r|p0.3|p0.3| Op & Address & Value & Comment
Write & & new value &
Write & & = 2, , , = 0x300 & Write `mstatus`

17.2.6.2. Using Program Buffer

Abstract commands are used to exchange data with GPRs. Using this mechanism, other registers can be accessed by moving their value into/out of GPRs.

Write `mstatus` using program buffer:

|c|r|p0.3|p0.3| Op & Address & Value & Comment
Write & & `csrw s0, MSTATUS` &
Write & `progbuf1` & `ebreak` &
Write & & new value &
Write & & = 2, , , , = 0x1008 & Write `s0`, then execute program buffer

Read `f1` using program buffer:

|c|r|p0.3|p0.3| Op & Address & Value & Comment
Write & & `fmv.x.s s0, f1` &
Write & `progbuf1` & `ebreak` &
Write & & & Execute program buffer
Write & & , = 0x1008 & read `s0`
Read & & - & Returns the value that was in `f1`

17.2.7. Reading Memory

17.2.7.1. Using System Bus Access

With system bus access, addresses are physical system bus addresses.

Read a word from memory using system bus access:

|c|r|p0.3|p0.3| Op & Address & Value & Comment
Write & & = 2, & Setup
Write & & address &
Read & & - & Value read from memory

Read block of memory using system bus access:

|r|r|p13em|l| Op & Address & Value & Comment
Write & & = 2, , , & Turn on autoread and autoincrement
Write & & address & Writing address triggers read and increment
Read & & - & Value read from memory
Read & & - & Next value read from memory
... & ... & ... & ...
Write & & 0 & Disable autoread
Read & & - & Get last value read from memory.

17.2.7.2. Using Program Buffer

Through the Program Buffer, the hart performs the memory accesses. Addresses are physical or virtual (depending on and other system configuration).

Read a word from memory using program buffer:

|c|r|p0.3|p0.3| Op & Address & Value & Comment
Write & & `lw s0, 0(s0)` &
Write & `progbuf1` & `ebreak` &
Write & & address &
Write & & , , , = 0x1008 & Write `s0`, then execute program buffer
Write & & = 0x1008 & Read `s0`
Read & & - & Value read from memory

Read block of memory using program buffer:

|c|r|p0.3|p0.3| Op & Address & Value & Comment
Write & & `lw s1, 0(s0)` &
Write & `progbuf1` & `addi s0, s0, 4` &
Write & `progbuf2` & `ebreak` &
Write & & address &
Write & & , , , = 0x1008 & Write `s0`, then execute program buffer
Write & & , = 0x1009 & Read `s1`, then execute program buffer
Write & & & Set
Read & & - & Get value read from memory, then execute program buffer
Read & & - & Get next value read from memory, then execute program buffer
... & ... & ... & ...
Write & & 0 & Clear

Read & & - & Get last value read from memory.

17.2.7.3. Using Abstract Memory Access

Abstract memory accesses act as if they are performed by the hart, although the actual implementation may differ.

Read a word from memory using abstract memory access:

|c|r|p0.3|p0.3| Op & Address & Value & Comment
Write & `data1`& address &
Write & & cmdtype=2, &
Read & & - & Value read from memory

Read block of memory using abstract memory access:

|c|r|p0.3|p0.3| Op & Address & Value & Comment
Write & & 1 & Re-execute the command when is accessed
Write & `data1`& address &
Write & & cmdtype=2, , &
Read & & - & Read value, and trigger reading of next address
... & ... & ... & ...
Write & & 0 & Disable auto-exec
Read & & - & Get last value read from memory.

17.2.8. Writing Memory

17.2.8.1. Using System Bus Access

With system bus access, addresses are physical system bus addresses.

Write a word to memory using system bus access:

|c|r|p0.3|p0.3| Op & Address & Value & Comment
Write & & = 2 & Configure access size
Write & & address &
Write & & value &

Write a block of memory using system bus access:

|c|r|p0.3|p0.3| Op & Address & Value & Comment
Write & & = 2, & Turn on autoincrement
Write & & address &
Write & & value0 &
Write & & value1 &
... & ... & ... & ...
Write & & valueN &

17.2.8.2. Using Program Buffer

Through the Program Buffer, the hart performs the memory accesses. Addresses are physical or virtual (depending on and other system configuration).

Write a word to memory using program buffer:

|c|r|p0.3|p0.3| Op & Address & Value & Comment
Write & & `sw s1, 0(s0)` &
Write & `progbuf1` & `ebreak` &
Write & & address &
Write & & , , = 0x1008 & Write `s0`
Write & & value &
Write & & , , , = 0x1009 & Write `s1`, then execute program buffer

Write block of memory using program buffer:

|c|r|p0.3|p0.3| Op & Address & Value & Comment
Write & & `sw s1, 0(s0)` &
Write & `progbuf1` & `addi s0, s0, 4` &
Write & `progbuf2` & `ebreak` &
Write & & address &
Write & & , , = 0x1008 & Write `s0`
Write & & value0 &
Write & & , , , = 0x1009 & Write `s1`, then execute program buffer
Write & & & Set
Write & & value1 &
... & ... & ... & ...
Write & & valueN &
Write & & 0 & Clear

17.2.8.3. Using Abstract Memory Access

Abstract memory accesses act as if they are performed by the hart, although the actual implementation may differ.

Write a word to memory using abstract memory access:

|c|r|p0.3|p0.3| Op & Address & Value & Comment
Write & `data1`& address &
Write & & value &
Write & & cmdtype=2, , write=1 &

Write a block of memory using abstract memory access:

|c|r|p0.3|p0.3| Op & Address & Value & Comment
Write & `data1`& address &
Write & & value0 &
Write & & cmdtype=2, , write=1, &
Write & & 1 & Re-execute the command when is accessed
Write & & value1 &
Write & & value2 &
... & ... & ... & ...
Write & & valueN &
Write & & 0 & Disable auto-exec

17.2.9. Triggers

A debugger can use hardware triggers to halt a hart when a certain event occurs. Below are some examples, but as there is no requirement on the number of features of the triggers implemented by a hart, these examples might not be applicable to all implementations. When a debugger wants to set a trigger, it writes the desired configuration, and then reads back to see if that configuration is supported. All examples assume XLEN=32.

Enter Debug Mode when the instruction at 0x80001234 is executed, to be used as an instruction breakpoint in ROM:

|r|r|L| & 0x6980105c & type=6, dmode=1, action=1, select=0, match=0, m=1, s=1, u=1, vs=1, vu=1, execute=1
& 0x80001234 & address

Enter Debug Mode when performing a load at address 0x80007f80 in M-mode or S-mode or U-mode:

|r|r|L| & 0x68001059 & type=6, dmode=1, action=1, select=0, match=0, m=1, s=1, u=1, load=1
& 0x80007f80 & address

Enter Debug Mode when storing to an address between 0x80007c80 and 0x80007cef (inclusive) in VS-mode or VU-mode when hgatp.VMID=1:

|r|r|L| & 0x69801902 & type=6, dmode=1, action=1, chain=1, select=0, match=2, vs=1, vu=1, store=1
& 0x80007c80 & start address (inclusive)
& 0x03000000 & mhselect=6, mhvalue=0
& 0x69801182 & type=6, dmode=1, action=1, select=0, match=3, vs=1, vu=1, store=1
& 0x80007cf0 & end address (exclusive)
& 0x03000000 & mhselect=6, mhvalue=0

Enter Debug Mode when storing to an address between 0x81230000 and 0x8123ffff (inclusive):

|r|r|L| & 0x698010da & type=6, dmode=1, action=1, select=0, match=1, m=1, s=1, u=1, vs=1, vu=1, store=1
& 0x81237fff & 16 upper bits to match exactly, then 0, then all ones.

Enter Debug Mode when loading from an address between 0x86753090 and 0x8675309f or between 0x96753090 and 0x9675309f (inclusive):

|r|r|L| & 0x69801a59 & type=6, dmode=1, action=1, chain=1, match=4, m=1, s=1, u=1, vs=1, vu=1, load=1
& 0xfff03090 & Mask for low half, then match for low half
& 0x698012d9 & type=6, dmode=1, action=1, match=5, m=1, s=1, u=1, vs=1, vu=1, load=1
& 0xefff8675 & Mask for high half, then match for high half

17.2.10. Handling Exceptions

Generally the debugger can avoid exceptions by being careful with the programs it writes. Sometimes they are unavoidable though, e.g. if the user asks to access memory or a CSR that is not implemented. A typical debugger will not know enough about the hardware platform to know what's going to happen, and must attempt the access to determine the outcome.

When an exception occurs while executing the Program Buffer, becomes set. The debugger can check this field to see whether a program encountered an exception. If there was an exception, it's left to the

debugger to know what must have caused it.

17.2.11. Quick Access

There are a variety of instructions to transfer data between GPRs and the ` data` registers. They are either loads/stores or CSR reads/writes. The specific addresses also vary. This is all specified in . The examples here use the pseudo-op `transfer dest, src` to represent all these options.

Halt the hart for a minimum amount of time to perform a single memory write:

|c|r|p0.3|p0.3| Op & Address & Value & Comment
Write & & `transfer arg2, s0` & Save `s0`
Write & `progbuf1` & `transfer s0, arg0` & Read first argument (address)
Write & `progbuf2` & `transfer arg0, s1` & Save `s1`
Write & `progbuf3` & `transfer s1, arg1` & Read second argument (data)
Write & `progbuf4` & `sw s1, 0(s0)` &
Write & `progbuf5` & `transfer s1, arg0` & Restore `s1`
Write & `progbuf6` & `transfer s0, arg2` & Restore `s0`
Write & `progbuf7` & `ebreak` &
Write & & address &
Write & `data1` & data &
Write & & 0x10000000 & Perform quick access

This shows an example of setting the bit in to enable a hardware breakpoint in M-mode. Similar quick access instructions could have been used previously to configure the trigger that is being enabled here:

|c|r|p0.3|p0.3| Op & Address & Value & Comment
Write & & `transfer arg0, s0` & Save `s0`
Write & `progbuf1` & `li s0, (1 << 6)` & Form the mask for bit
Write & `progbuf2` & `csrrs x0, , s0` & Apply the mask to
Write & `progbuf3` & `transfer s0, arg2` & Restore `s0`
Write & `progbuf4` & `ebreak` &
Write & & 0x10000000 & Perform quick access

17.3. Native Debugger Implementation

The spec contains a few features to aid in writing a native debugger. This section describes how some common tasks might be achieved.

17.3.1. Single Step

Single step is straightforward if the OS or a debug stub runs in M-Mode while the program being debugged runs in a less privileged mode. When a step is required, the OS or debug stub writes , , before returning control to the lower user program with an `mret` instruction.

Stepping code running in the same privilege mode as the debugger is more complicated, depending on what other debug features are implemented.

If hardware implements and , then stepping through non-trap code which doesn't allow for nested interrupts is also straightforward.

If hardware automatically prevents triggers from matching when entering a trap handler as described

in Section #sec:nativetrigger[6.4], then a carefully written trap handler can ensure that interrupts are disabled whenever the icount trigger must not match.

If neither of these features exist, then single step is doable, but tricky to get right. To single step, the debug stub would execute something like:

```
    li    t0, \FcsrIcountCount=4, \FcsrIcountAction=0, \FcsrIcountM=1
    csrw  tdata1, t0    /* Write the trigger. */
    lw    t0, 8(sp)     /* Restore t0, count decrements to 3 */
    lw    sp, 0(sp)     /* Restore sp, count decrements to 2 */
    mret                /* Return to program being debugged. count decrements to 1 */
```

There is an additional problem with using to single step. An instruction may cause an exception into a more privileged mode where the trigger is not enabled. The exception handler might address the cause of the exception, and then restart the instruction. Examples of this include page faults, FPU instructions when the FPU is not yet enabled, and interrupts. When a user is single stepping through such code, they will have to step twice to get past the restarted instruction. The first time the exception handler runs, and the second time the instruction actually executes. That is confusing and usually undesirable.

To help users out, debuggers should detect when a single step restarted an instruction, and then step again. This way the users see the expected behavior of stepping over the instruction. Ideally the debugger would notify the user that an exception handler executed the first time.

The debugger should perform this extra step when the PC doesn't change during a regular step.

> — —
>
> ```
>  +
> It is safe to perform an extra step when the PC changes, because every
> RISC-V instruction either changes the PC or has side effects when
> repeated, but never both.
> ```

To avoid an infinite loop if the exception handler does not address the cause of the exception, the debugger must execute no more than a single extra step.

[index]#

Chapter 18. Change Log

Chapter 19. Future Ideas

All items in this section are future ideas and should not be considered part of the specification.

Some future version of this spec may implement some of the following features.

1. The spec defines several additions to the Device Tree which enable a debugger to discover hart IDs and supported triggers for all the harts in the system.
2. DTMs can function as general bus subordinates, so they would look like regular RAM to bus managers.
3. Harts can be divided into groups. All the harts in the same group can be halted/run/stepped simultaneously. When a hart hits a breakpoint, all the other harts in the same group also halt within a few clock cycles.
4. DTMs are specified for protocols like USB, I2C, SPI, and SWD.
5. The debugger can communicate with the power manager to power cores up or down, and to query their status.
6. Serial ports can raise an interrupt when a send/receive queue becomes full/empty.
7. The debug interrupt can be masked by running code. If the interrupt is asserted, then deasserted, and then asserted again the debug interrupt happens anyway. This mechanism can be used to e.g. read/write memory with minimal interruption, making sure never to interrupt during a critical piece of code.
8. The Debug Module can include a serial interface for re-using the DTM interface as a generic communication interface.

19.1. Serial Ports

The Debug Module may implement up to 8 serial ports. They support basic flow control and full duplex data transfer between a component and the debugger, essentially allowing the Debug Transport to be used to communicate with a debug monitor running on a hart, or more generally emulate devices which aren't present. All these uses require software support, and are not further specified here. Only the DMI side of the Debug Module serial registers are defined in this specification as the core side interface should look like a peripheral device.